importing libraries

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

Loading dataset

```
df=pd.read_csv('https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv')
df.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	durati
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 m
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	Seaso
					Sami Bouajila,					

Next steps: Generate code with df View recommended plots

Shape of dataframe

```
df.shape
```

```
(8807, 12)
```

Comment : Dataframe contains 8807 rows and 12 Columns.

Information of DataFrame

```
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
```

Checkin unique values per column

```
df.nunique()
```

```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1767
release_year      74
rating            17
duration         220
listed_in        514
description     8775
dtype: int64
```

Statistical summary

```
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

```
df.describe(include="object")
```

	show_id	type	title	director	cast	country	date_added	rating	duration	list
count	8807	8807	8807	6173	7982	7976	8797	8803	8804	
unique	8807	2	8807	4528	7692	748	1767	17	220	
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	TV-MA	1 Season	Dr Intern

Checking Duplicates

```
df.duplicated().sum()
```

```
0
```

Checking number of Null values

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

```
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
```

Comment: Director Column contains 2634 null values and cast contains 825 and country contains 831 Null values.

Handling missing values:

For director and cast we are replacing missing values with 'NO Data' for data intigrity

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

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

For Country column we will use mod to 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

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

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]


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)
```

Droping the remaning values

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

```
(8797, 12)
```

Conversion of data type

```
df["date_added"] =  df['date_added'].apply(lambda x: pd.to_datetime(x).strftime('%d/%m/%Y'))
df['date_added']=pd.to_datetime(df['date_added'])
df['date_added']
```

```
<ipython-input-16-a55b1d498ded>:2: UserWarning: Parsing dates in %d/%m/%Y format when dayfirst=False (the default) was specified. Pass `dayfirst=True` o
  df['date_added']=pd.to_datetime(df['date_added'])
0      2021-09-25
1      2021-09-24
2      2021-09-24
3      2021-09-24
4      2021-09-24
          ...
8802   2019-11-20
8803   2019-07-01
8804   2019-11-01
8805   2020-01-11
8806   2019-03-02
Name: date_added, Length: 8797, dtype: datetime64[ns]
```

```
df['Month']=df['date_added'].dt.month
df['Month_name']=df['date_added'].dt.month_name()
df["year"]=df['date_added'].dt.year
df.head(5)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duratio
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	No Data	United States	2021-09-25	2020	PG-13	90 m
1	s2	TV Show	Blood & Water	No Data	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	Season
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	2021	TV-MA	1 Seas

Next steps: Generate code with df View recommended plots

Unnseting of data :

```
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_cast
```

	cast	show_id
0	No Data	s1
1	Ama Qamata	s2
1	Khosi Ngema	s2
1	Gail Mabalane	s2
1	Thabang Molaba	s2
...	...	...
8806	Manish Chaudhary	s8807
8806	Meghna Malik	s8807
8806	Malkeet Rauni	s8807
8806	Anita Shabdish	s8807
8806	Chittaranjan Tripathy	s8807

64882 rows × 2 columns

Next steps: Generate code with df_cast View recommended plots

```
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']
df_director
```

	director	show_id
0	Kirsten Johnson	s1
1	No Data	s2
2	Julien Leclercq	s3
3	No Data	s4
4	No Data	s5
...	...	...
8802	David Fincher	s8803
8803	No Data	s8804
8804	Ruben Fleischer	s8805
8805	Peter Hewitt	s8806
8806	Mozez Singh	s8807

9602 rows × 2 columns

Next steps: Generate code with df_director View recommended plots

```
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_country
```

	country	show_id
0	United States	s1
1	South Africa	s2
2	United States	s3
3	United States	s4
4	India	s5
...	...	...
8802	United States	s8803
8803	United States	s8804
8804	United States	s8805
8805	United States	s8806
8806	India	s8807

10840 rows × 2 columns

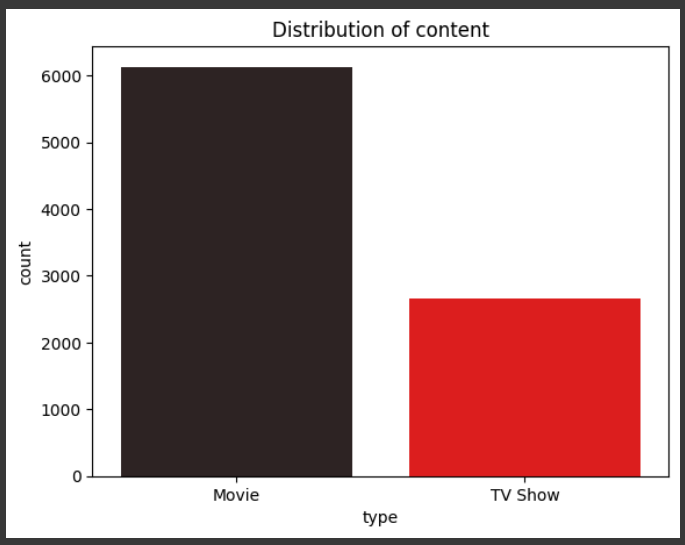
Next steps: Generate code with df_country View recommended plots

Creating dataset contains only Movies and TV shows

```
Movies=df.loc[df['type']=='Movie']
Shows=df.loc[df['type']=='TV Show']
```

Distrbution of Content type

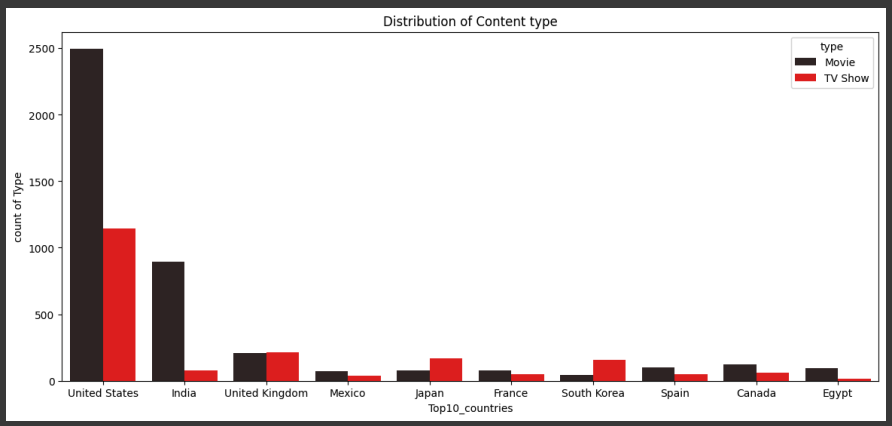
```
sns.countplot(x='type',data=df,hue='type',palette='dark:red')
plt.title("Distribution of content")
plt.show()
```



Comments: Above countplot visiualization shows that around 6000 countires contains movie content and around 3000 countries contains TV Shows contents

Distribution of content type in different countries:

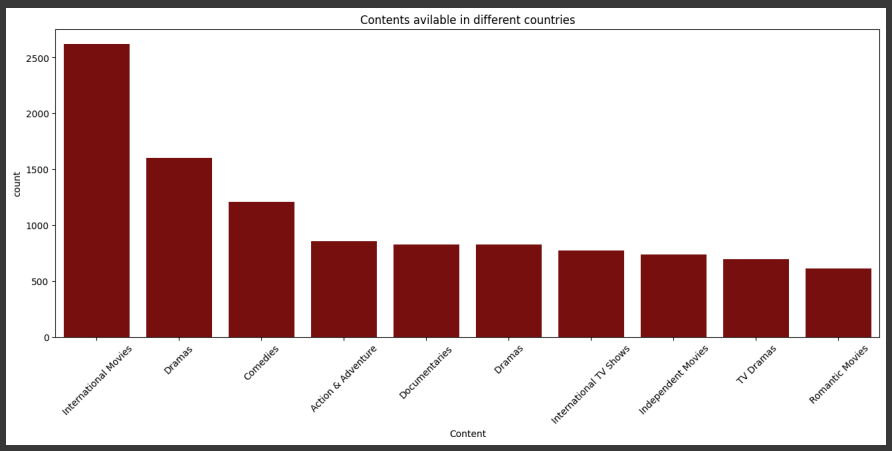
```
top10=df['country'].value_counts().index[:10]
Content=df.loc[df['country'].isin(top10)]
plt.figure(figsize=(14,6))
sns.countplot(x='country',hue='type',data=Content,palette="dark:red")
plt.title("Distribution of Content type")
plt.xlabel("Top10_countries")
plt.ylabel("count of Type")
plt.show()
```



Comments: In top 10 countris US has highest Movie content and also it has highest number of TV Shows,followed by India, UK and lastly Egypt has very low number of Movie and TV SHows contents.

Content avilable in different Countries

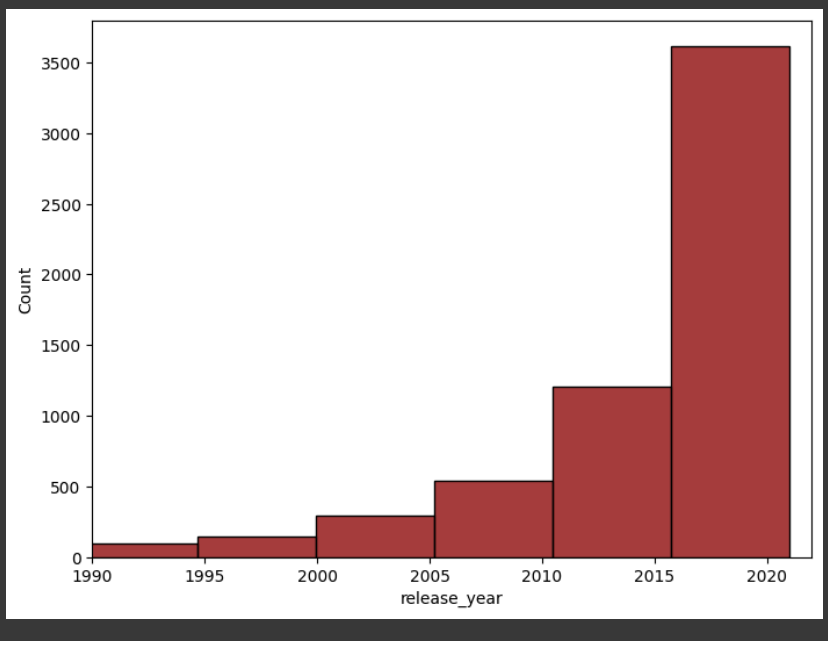
```
df_cont = df['listed_in'].str.split(',', expand=True).stack()
df_cont = df_cont.reset_index(level=1, drop=True).to_frame('listed_in')
df_cont[['show_id','country']] = df[['show_id','country']]
topc_10=df_cont['listed_in'].value_counts().head(10)
plt.figure(figsize=(16,6))
sns.barplot(x=topc_10.index,y=topc_10.values,color='darkred')
plt.title("Contents avilable in different countries")
plt.xlabel("Content")
plt.ylabel("count")
plt.xticks(rotation=45)
plt.show()
```



Comment: above bar chart shows that International Movies content has more dominant than other contents.

Number of movies released last 20 to 30 years

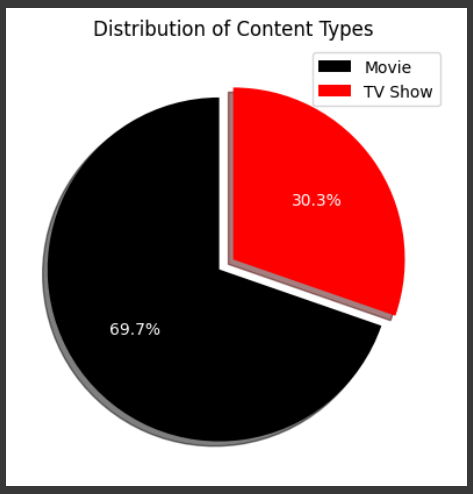
```
plt.figure(figsize=(8,6))
sns.histplot(Movies['release_year'],color='darkred',bins=15)
plt.xlim(1990,2022)
plt.show()
```



Comments: The above histogram visualization shows that the number of movies released last 30 years increased and we can see that from last 5 years around 3500 movies are released.

Comparision of TV Shows vs Movies

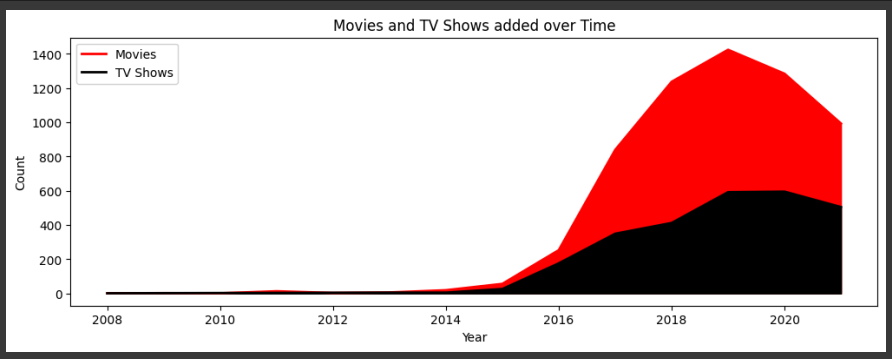
```
x=df.groupby('type')['type'].count()
y=len(df)
r=(x/y*100).round(2)
mf_ratio = pd.DataFrame(r)
mf_ratio.rename({'type': '%'}, axis=1, inplace=True)
plt.pie(mf_ratio['%'], labels=mf_ratio.index, autopct='%1.1f%%', colors=['black','red'],
 explode=(0.1,0), shadow=True, startangle=90,textprops={'color': 'white'}
)
plt.legend(loc='upper right')
plt.title('Distribution of Content Types')
plt.show()
```



Comments: The pie chart visiualzation shows that Netflix consists apporximatly 70% Movies and other hand 30% TV Shows.

Does Netflix has more focous on TV Shows added in recent Years.

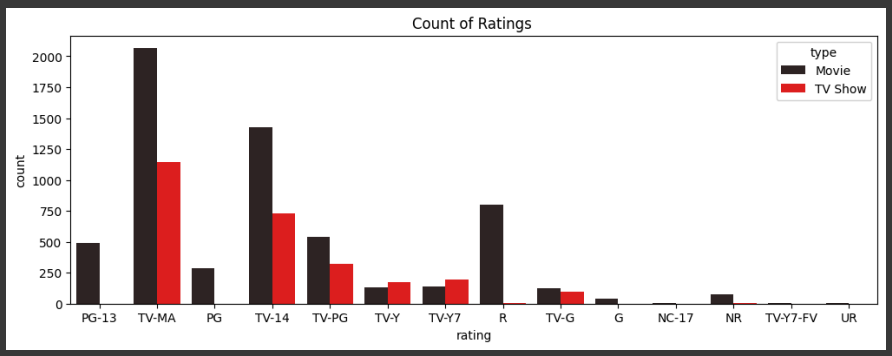
```
Movies_cnt=Movies['year'].value_counts().sort_index()
Shows_cnt=Shows['year'].value_counts().sort_index()
plt.figure(figsize=(12,4))
plt.plot(Movies_cnt.index,Movies_cnt.values,color='red',label="Movies",linewidth=2)
plt.plot(Shows_cnt.index,Shows_cnt.values,color="black",label="TV Shows",linewidth=2)
plt.fill_between(Movies_cnt.index,Movies_cnt.values,color="red")
plt.fill_between(Shows_cnt.index,Shows_cnt.values,color="black")
plt.xlabel("Year")
plt.ylabel("Count")
plt.title("Movies and TV Shows added over Time")
plt.legend()
plt.show()
```



Comments: The above plot shows that Netflix added more number of Movies than the TV Shows over the Years.

Distribution of Ratings

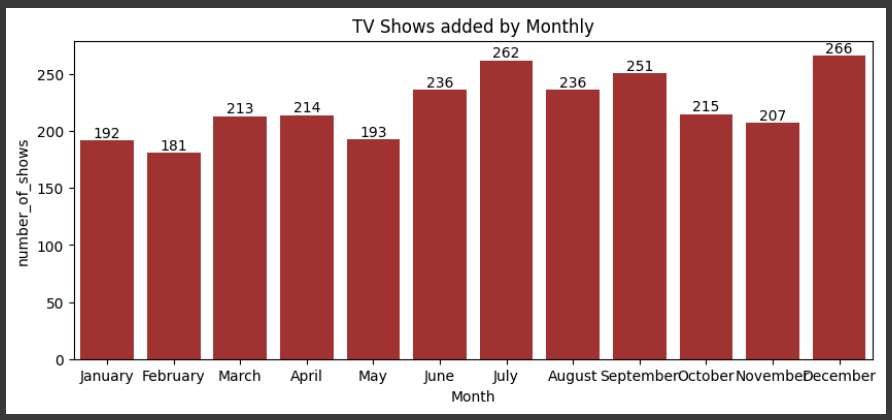
```
cr=df.drop(df[df['rating']=='74 min'].index)
cr.drop(cr[cr['rating']=='84 min'].index,inplace=True)
cr.drop(cr[cr['rating']=='66 min'].index,inplace=True)
plt.figure(figsize=(12,4))
sns.countplot(x='rating',hue='type',data=cr,palette='dark:red')
plt.title("Count of Ratings ")
plt.show()
```



Comments:Countplot visiualization shows that majority of movies and TV shows falls into TV-MA and TV-14 ratings,and TV-Y, TV-Y7 ratings are higher for TV shows than the movies.

Shows added by Monthly

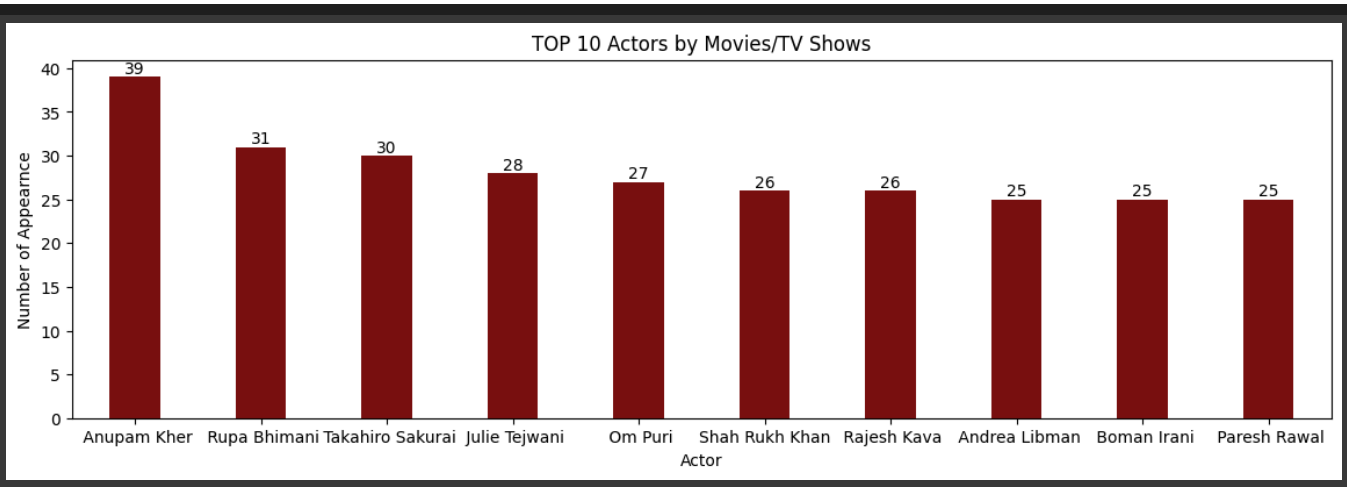
```
month_order=['January', 'February', 'March', 'April', 'May', 'June', 'July',
               'August', 'September', 'October', 'November', 'December']
monthly_count=Shows['Month_name'].value_counts().loc[month_order]
plt.figure(figsize=(10,4))
plot=sns.barplot(x=monthly_count.index,y=monthly_count.values,color="firebrick")
for index, value in enumerate(monthly_count.values):
    plot.text(index, value, str(value), ha='center', va='bottom')
plt.xlabel("Month")
plt.ylabel("number_of_shows")
plt.title("TV Shows added by Monthly")
plt.show()
```



Comment:Above bar plot shows that netflix most of the shows added in july and december.So it's show that July or December is best time to launch TV show bcz viewers who want to anticipate new releases during these months.

Actors by Movies/TV Shows

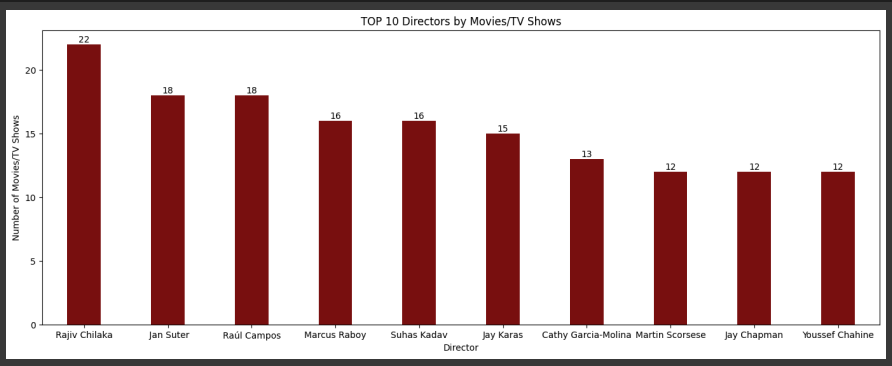
```
actor_counts=df_cast['cast'].value_counts()[1:]
Top_actors=actor_counts.head(10)
plt.figure(figsize=(14,4))
bar_plot=sns.barplot(x=Top_actors.index,y=Top_actors.values,color='darkred',width=0.4)
plt.title("TOP 10 Actors by Movies/TV Shows")
plt.xlabel("Actor")
plt.ylabel("Number of Appearnce")
for index, value in enumerate(Top_actors.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')
plt.show()
```



Comments:In top10 actors the above barplots shows that "Anupam Kher" has the highest appeareance in Movies/TV Shows.

Directors by Movies/TV Shows

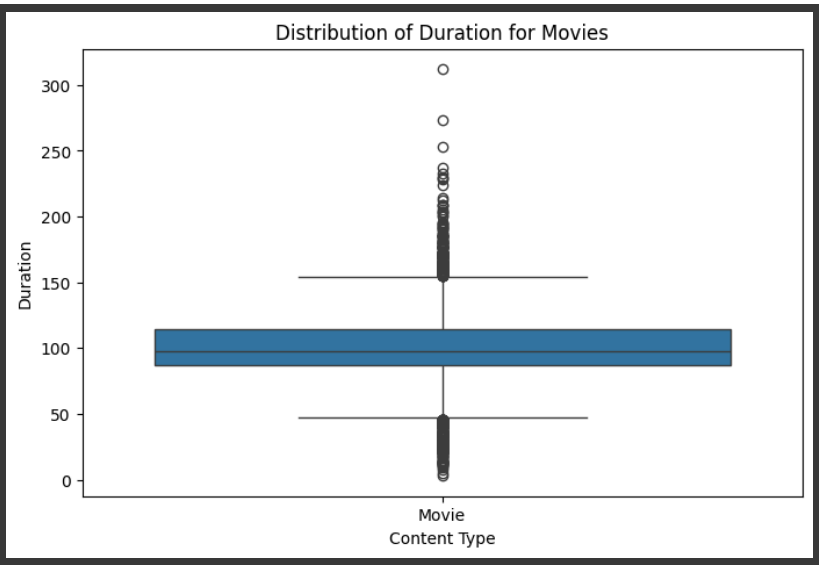
```
director_counts=df_director['director'].value_counts()[1:]
Top_director=director_counts.head(10)
plt.figure(figsize=(17,6))
bar_plot=sns.barplot(x=Top_director.index,y=Top_director.values,color='darkred',width=0.4)
plt.title("TOP 10 Directors by Movies/TV Shows")
plt.xlabel("Director")
plt.ylabel("Number of Movies/TV Shows")
for index, value in enumerate(Top_director.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')
plt.show()
```



Comments: The above barplot shows that "Rajiv Chilaka" has directed highest number of Movies/Shows

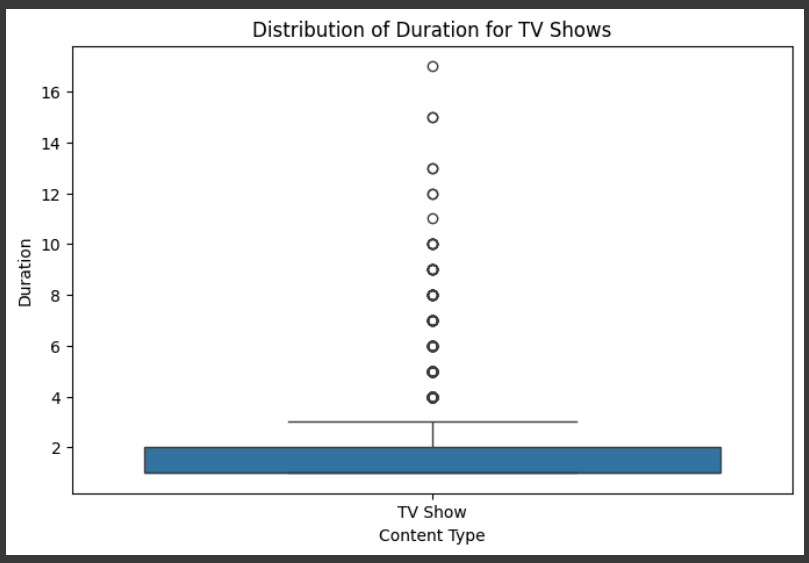
Duration Distribution for Movies and TV Shows

```
Md=Movies.copy()
Md['duration'] = Md['duration'].str.extract('(\d+)', expand=False).astype(int)

# Creating a boxplot for movie duration
plt.figure(figsize=(8, 5))
sns.boxplot(data=Md, x='type', y='duration')
plt.xlabel('Content Type')
plt.ylabel('Duration')
plt.title('Distribution of Duration for Movies')
plt.show()
```



Comments: Above boxplot for Movies shows that most of Movies fall within the resonable duration.few outliers are exceeding resonable duration.

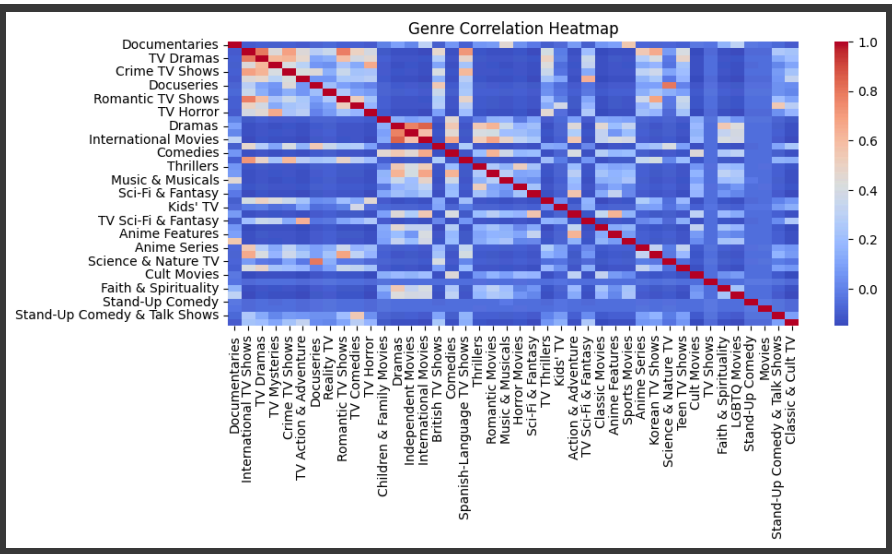
```
Tshows=Shows.copy()
Tshows['duration'] = Tshows['duration'].str.extract('(\d+)', expand=False).astype(int)

# Creating a boxplot for TV show duration
plt.figure(figsize=(8, 5))
sns.boxplot(data=Tshows, x='type', y='duration')
plt.xlabel('Content Type')
plt.ylabel('Duration')
plt.title('Distribution of Duration for TV Shows')
plt.show()
```



Comments: Above boxplot for TV-Shows shows that most of the shows are fall within the 2 to 4 Seasons but, few outliers are having more than 6 seasons.

HeatMap

```
# Extracting unique genres from the 'listed_in' column
genres = df['listed_in'].str.split(', ', expand=True).stack().unique()
genre_data = pd.DataFrame(index=genres, columns=genres, dtype=float)
genre_data.fillna(0, inplace=True)
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
correlation_matrix = genre_data.corr()
plt.figure(figsize=(10, 4))
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm')
plt.title('Genre Correlation Heatmap')
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.show()
```



Business Insights :

1. From Movies and shows added plots we come to know that Netflix Started significant growth from 2015.

2.From monthly added plots we come to know that Neflix will add more Shows in july and december.

3.Most of the contents types are avilable on Netflix are Movies.

4.United States are highest content produced yearly.

5.Most of the Contents are fall under TV-MA ratings.

6.Most Netflix content genre fall under Dramas, Comedies and Action & Adventure.

Recommendations :

1. From the dataset we came to know that US and India has contains highest Content type so Netflix has to understand that what other countries Peoples are interested in the contents so it has to comeup with an idea to attract other countries.

2. From data analysis we came to know that last 20 to 30 years Netflix has more focoused on Movies.So Netflix can add more number of TV Shows because some of the countries are interested in TV Shows.

3.In Most of the countries International Movies,Drama,Comidies content is aviable so Netflix can add more number of Movies/Tv Shows releated to Drama and comidies.

```
Start coding or generate with AI.
```