```
from google.colab import files
uploaded = files.upload()
```

Choose Files Netflix Dataset (1).csv

- **Netflix Dataset (1).csv**(text/csv) - 2962357 bytes, last modified: 8/3/2025 - 100% done

```
Saving Netflix Dataset (1).csv to Netflix Dataset (1).csv
```

```
import pandas as pd

# Replace 'your_file.csv' with your actual filename
df = pd.read_csv('Netflix Dataset (1).csv')

# Preview first few rows
df.head()
```

	Show_Id	Category	Title	Director	Cast	Country	Release_Date	Rating	Duration	Type	Description
0	s1	TV Show	3%	NaN	João Miguel, Bianca Comparato, Michel Gomes, R...	Brazil	August 14, 2020	TV-MA	4 Seasons	International TV Shows, TV Dramas, TV Sci-Fi &...	In a future where the elite inhabit an island ...
1	s2	Movie	07:19	Jorge Michel Grau	Demián Bichir, Héctor Bonilla, Oscar Serrano, ...	Mexico	December 23, 2016	TV-MA	93 min	Dramas, International Movies	After a devastating earthquake hits Mexico Cit...
2	s3	Movie	23:59	Gilbert Chan	Tedd Chan, Stella Chung, Henley Hii,	Singapore	December 20, 2018	R	78 min	Horror Movies, International	When an army recruit is found

Next steps: Generate code with df View recommended plots New interactive sheet

Start coding or generate with AI.

Task

Perform a comprehensive data analysis on the provided CSV file using Python. The analysis should include data loading, handling missing values, descriptive statistics, various visualizations (bar charts, pie charts, line charts, correlation heatmap, histograms), analysis of patterns, trends, and outliers, and a summary of 5 key insights. Ensure the output is visually appealing and well-structured.

Data loading and initial inspection

Subtask:

Load the dataset into a pandas DataFrame and display the first few rows, along with information about the columns and data types.

Reasoning: Display the first few rows and the information about the DataFrame to understand its structure and data types.

```
df.head()
df.info()
```

```
<class 'pandas.core.frame.Da
RangeIndex: 7789 entries, 0
Data columns (total 11 colum
 #   Column        Non-Null
---  ------        ---------
 0   Show_Id       7789 non-
 1   Category      7789 non-
 2   Title         7789 non-
 3   Director      5401 non-
 4   Cast          7071 non-
 5   Country       7282 non-
 6   Release_Date  7779 non-
 7   Rating        7782 non-null   object
 8   Duration      7789 non-null   object
 9   Type          7789 non-null   object
```

✓ Data loading and initial inspection
✓ Handle missing values
✓ Descriptive statistics
✓ Data visualization
✓ Analyze patterns, trends, and outliers
✓ Key insights
✓ Finish task

What can I help you build?

Gemini can make mistakes so double-check it and use code with caution. Learn more

```
 10  Description   7789 non-null   object
dtypes: object(11)
memory usage: 669.5+ KB
```

Handle missing values

Subtask:

Identify and handle missing values in the dataset, potentially by imputation or dropping rows/columns, depending on the extent of missingness.

Reasoning: Calculate and display the number of missing values for each column to understand the extent of missingness.

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

	0
Show_Id	0
Category	0
Title	0
Director	2388
Cast	718
Country	507
Release_Date	10
Rating	7
Duration	0
Type	0
Description	0

dtype: int64

Reasoning: Handle missing values based on the calculated counts and verify the result. 'Director', 'Cast', and 'Country' have a significant number of missing values and will be filled with a placeholder. 'Release_Date' and 'Rating' have very few missing values and the rows with these missing values will be dropped as they are important columns for analysis.

```
# Fill missing 'Director', 'Cast', and 'Country' with 'Unknown'
df['Director'].fillna('Unknown', inplace=True)
df['Cast'].fillna('Unknown', inplace=True)
df['Country'].fillna('Unknown', inplace=True)

# Drop rows with missing 'Release_Date' and 'Rating'
df.dropna(subset=['Release_Date', 'Rating'], inplace=True)

# Verify missing values are handled
df.isnull().sum()
```

```
/tmp/ipython-input-3411469769.py:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assigr
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting valu

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].me


  df['Director'].fillna('Unknown', inplace=True)
/tmp/ipython-input-3411469769.py:3: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assigr
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting valu

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].me


  df['Cast'].fillna('Unknown', inplace=True)
/tmp/ipython-input-3411469769.py:4: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assigr
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting valu

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].me


  df['Country'].fillna('Unknown', inplace=True)
```

	0
Show_Id	0
Category	0
Title	0
Director	0
Cast	0
Country	0
Release_Date	0
Rating	0
Duration	0
Type	0
Description	0

dtype: int64

Descriptive statistics

Subtask:

Generate summary statistics for numerical columns and explore the distribution of categorical columns.

Reasoning: Generate descriptive statistics for all columns and explore the distribution of 'Category' and 'Rating' columns as instructed.

```
# Generate descriptive statistics for all columns
display(df.describe(include='all'))

# Explore the distribution of 'Category' column
display(df['Category'].value_counts())

# Explore the distribution of 'Rating' column
display(df['Rating'].value_counts())
```

	Show_Id	Category	Title	Director	Cast	Country	Release_Date	Rating	Duration	Type	Description
count	7772	7772	7772	7772	7772	7772	7772	7772	7772	7772	7772
unique	7770	2	7770	4049	6818	682	1563	14	216	491	7752
top	s6621	Movie	The Lost Okoroshi	Unknown	Unknown	United States	January 1, 2020	TV-MA	1 Season	Documentaries	Multiple women report their husbands as missin...
freq	2	5374	2	2375	718	2547	118	2863	1606	334	3

	count
Category	
Movie	5374
TV Show	2398

dtype: int64

	count
Rating	
TV-MA	2863
TV-14	1928
TV-PG	804
R	665
PG-13	386
TV-Y	279
TV-Y7	270
PG	247
TV-G	194
NR	83
G	39
TV-Y7-FV	6
UR	5

Data visualization

Subtask:

Create various visualizations as requested, including bar charts, pie charts, line charts, a correlation heatmap, and histograms for numerical features to explore patterns and relationships.

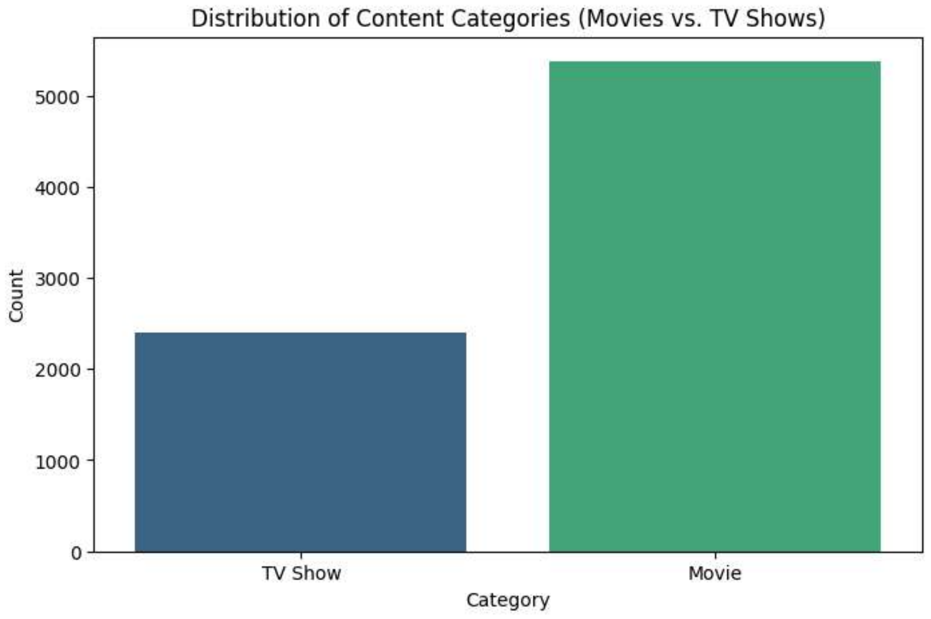
Reasoning: I will generate the requested visualizations step by step. First, I will create bar charts for content categories, ratings, top directors, and top actors. Then, I will create a pie chart for the top 10 countries. Finally, I will extract the release year and create a histogram and a bar chart for release year distributions and content type over the years.

```
import matplotlib.pyplot as plt
import seaborn as sns
import collections

# 1. Bar chart for content categories
plt.figure(figsize=(8, 5))
sns.countplot(data=df, x='Category', palette='viridis')
plt.title('Distribution of Content Categories (Movies vs. TV Shows)')
plt.xlabel('Category')
plt.ylabel('Count')
plt.show()

# 3. Bar chart for content ratings
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='Rating', order=df['Rating'].value_counts().index, palette='magma')
```

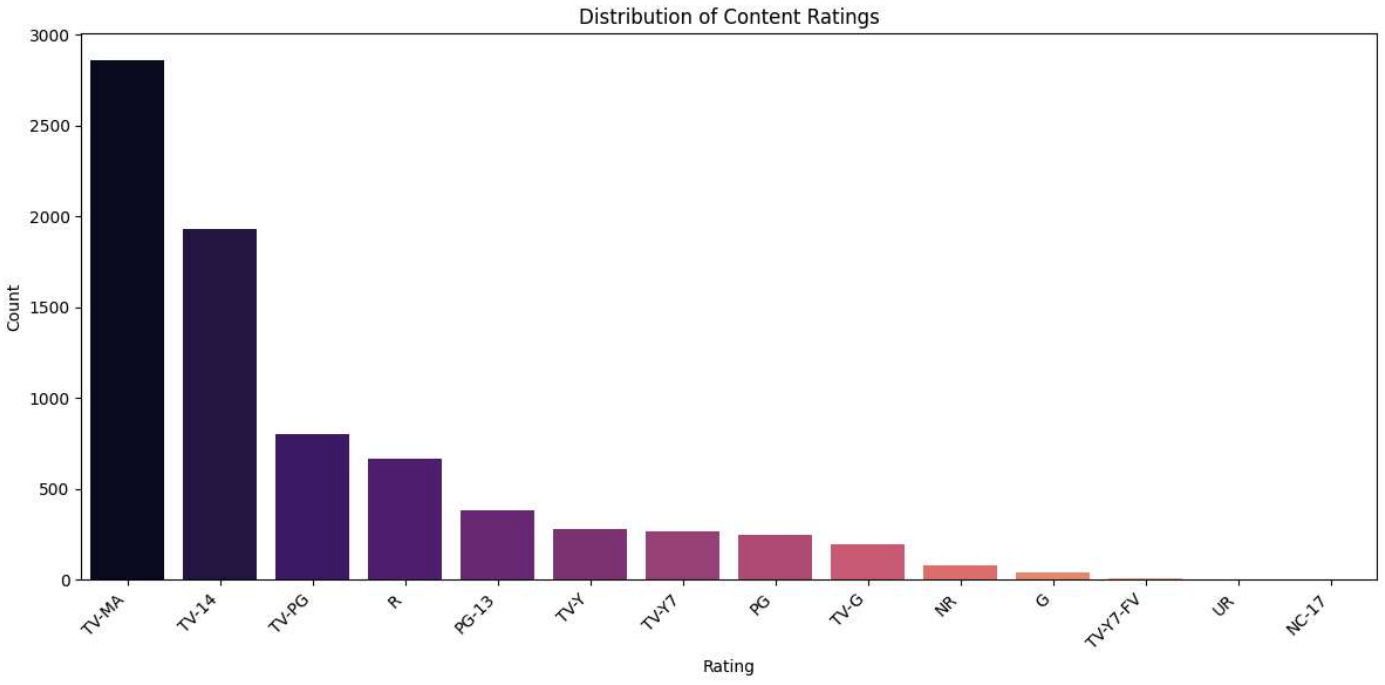
```
plt.title('Distribution of Content Ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# 6. Bar chart showing the top 10 directors
# Split directors and count
directors = df['Director'].str.split(', ').explode()
top_directors = directors.value_counts().head(11) # Get top 11 to potentially exclude 'Unknown' if it's the top
if 'Unknown' in top_directors.index:
    top_directors = top_directors.drop('Unknown').head(10)
else:
    top_directors = top_directors.head(10)

plt.figure(figsize=(12, 6))
sns.barplot(x=top_directors.index, y=top_directors.values, palette='plasma')
plt.title('Top 10 Directors with Most Content on Netflix')
plt.xlabel('Director')
plt.ylabel('Number of Titles')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# 7. Bar chart showing the top 10 actors
# Split actors and count
actors = df['Cast'].str.split(', ').explode()
top_actors = actors.value_counts().head(11) # Get top 11 to potentially exclude 'Unknown'
if 'Unknown' in top_actors.index:
    top_actors = top_actors.drop('Unknown').head(10)
else:
    top_actors = top_actors.head(10)

plt.figure(figsize=(12, 6))
sns.barplot(x=top_actors.index, y=top_actors.values, palette='viridis')
plt.title('Top 10 Actors with Most Content on Netflix')
plt.xlabel('Actor')
plt.ylabel('Number of Titles')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# 2. Pie chart for the proportion of the top 10 countries
# Count countries, handle multiple countries and 'Unknown'
countries = df['Country'].str.split(', ').explode()
country_counts = countries.value_counts()
# Exclude 'Unknown' if it's in the top
if 'Unknown' in country_counts.index:
    country_counts = country_counts.drop('Unknown')

top_10_countries = country_counts.head(10)

plt.figure(figsize=(10, 10))
plt.pie(top_10_countries, labels=top_10_countries.index, autopct='%1.1f%%', startangle=140, colors=sns.color_palette('Spectral', 10))
plt.title('Proportion of Top 10 Countries with Most Content on Netflix')
plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()

# Extract Release Year
df['Release_Year'] = pd.to_datetime(df['Release_Date'], errors='coerce').dt.year
# Drop rows where Release_Year could not be parsed
df.dropna(subset=['Release_Year'], inplace=True)
df['Release_Year'] = df['Release_Year'].astype(int)


# 4. Histogram for 'Release_Year'
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='Release_Year', bins=30, kde=True, palette='Blues')
plt.title('Distribution of Content Release Years')
plt.xlabel('Release Year')
plt.ylabel('Count')
plt.show()


# 5. Bar chart showing the number of movies and TV shows released over the years
release_year_category = df.groupby(['Release_Year', 'Category']).size().unstack().fillna(0)
```

```
plt.figure(figsize=(14, 7))
release_year_category.plot(kind='bar', stacked=True, figsize=(14, 7), colormap='viridis')
plt.title('Number of Movies and TV Shows Released Over the Years')
plt.xlabel('Release Year')
plt.ylabel('Number of Titles')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Category')
plt.tight_layout()
plt.show()

# 8. Correlation heatmap - Justification for skipping
print("A correlation heatmap is not suitable for this dataset as it primarily contains categorical data.")
print("There are no directly numerical features to calculate meaningful correlations between.")
```

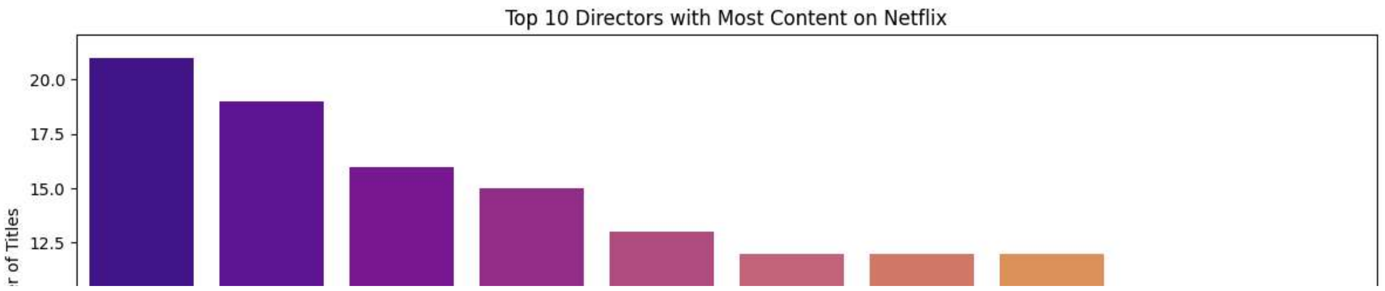
```
/tmp/ipython-input-3629525517.py:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.countplot(data=df, x='Category', palette='viridis')
```

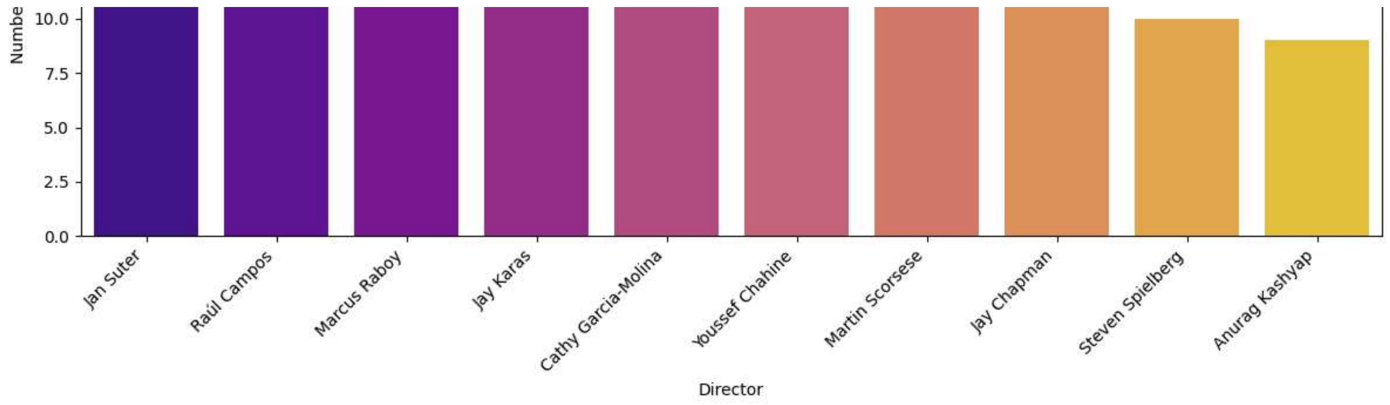


```
/tmp/ipython-input-3629525517.py:15: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.countplot(data=df, x='Rating', order=df['Rating'].value_counts().index, palette='magma')
```

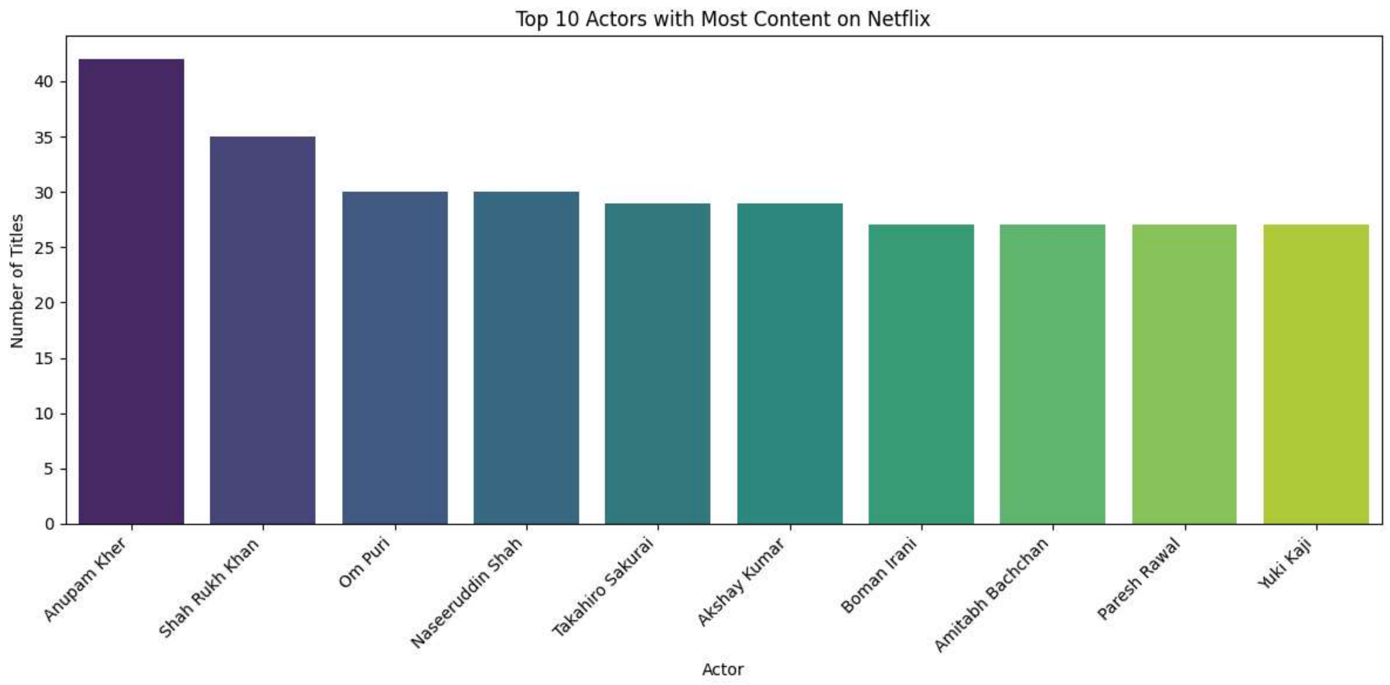


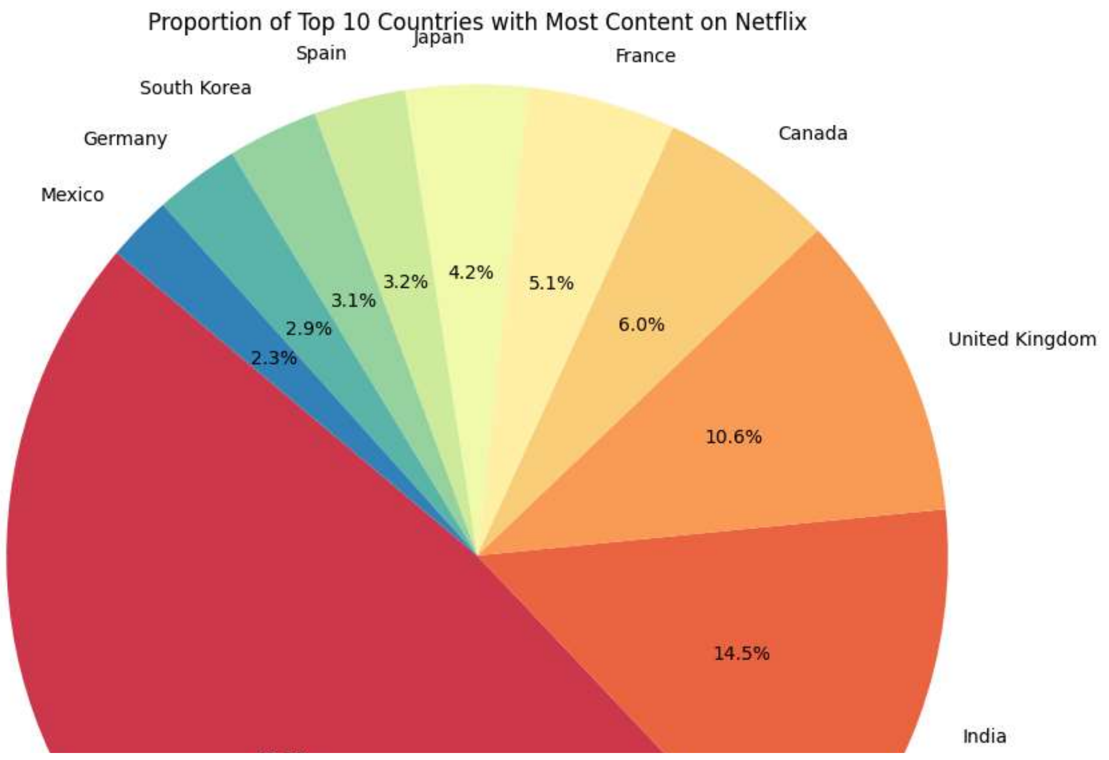
```
/tmp/ipython-input-3629525517.py:33: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.barplot(x=top_directors.index, y=top_directors.values, palette='plasma')
```





```
/tmp/ipython-input-3629525517.py:51: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.barplot(x=top_actors.index, y=top_actors.values, palette='viridis')
```


Top 10 Actors with Most Content on Netflix
Number of Titles
Anupam Kher
Shah Rukh Khan
Om Puri
Naseeruddin Shah
Takahiro Sakurai
Akshay Kumar
Boman Irani
Amitabh Bachchan
Paresh Rawal
Yuki Kaji
Actor


Proportion of Top 10 Countries with Most Content on Netflix
Japan
Spain
France
South Korea
Canada
Germany
Mexico
United Kingdom
India