Netflix Data Analysis: Unveiling Streaming Insights

This project explores Netflix data to extract actionable insights. We will use a data-driven approach. The goal is to understand user content trends. This will help improve Netflix's content strategy.

by Nimit Tiwari

Data Source: Kaggle Netflix Dataset

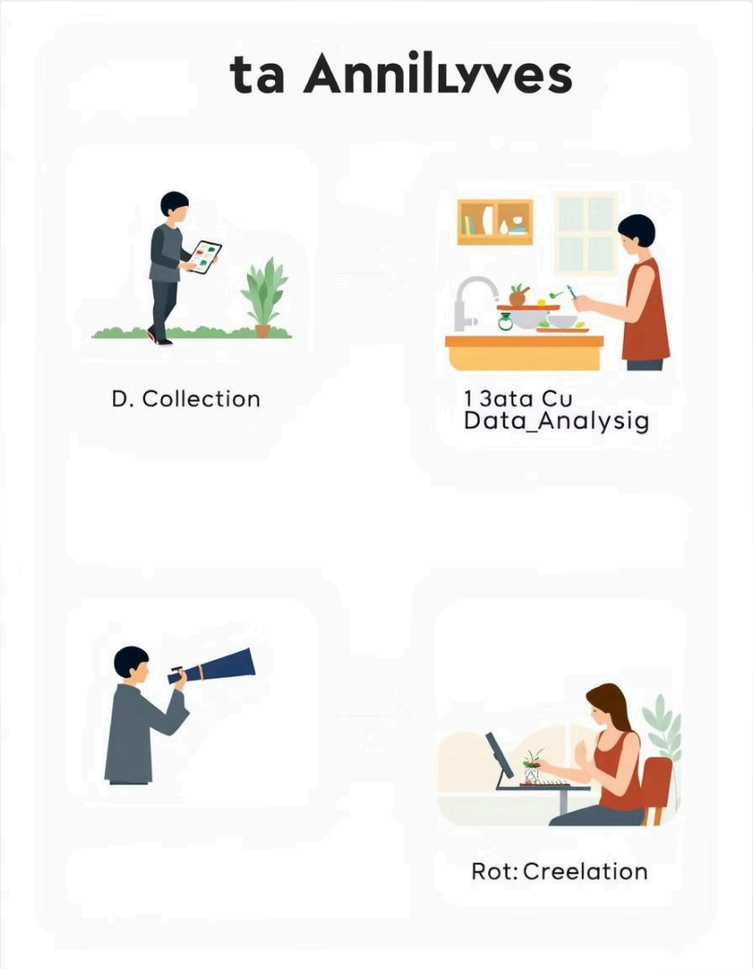
Netflix is a popular streaming service that offers a vast catalog of movies, TV shows, and original contents. The data comes from a publicly available Kaggle dataset. It is titled "Netflix1". The data consist of contents added to Netflix from 2008 to 2021. The oldest content is as old as 1925 and the newest as 2021. It contains roughly 8790 titles, including both movies and TV shows.

Variables

- Show id
- Type
- Title
- Director
- Country
- Date Added
- Release Year
- Rating
- Duration
- Listed In

Project Objectives

- This project involves loading, cleaning, analyzing, and visualizing data from a Netflix dataset.
- We'll use Python libraries like Pandas, Matplotlib, and Seaborn to work through the project.
- We'll analyze data trends and distributions using summary statistics and visualization in python.
- The goal is to explore the dataset, initial exploration to generating actionable insights.



Methodology: Data Analysis Workflow

Our analysis follows a structured workflow. Each step contributes to actionable insights. This workflow ensures a thorough and data-driven approach.

1

Import Required Libraries and Load the Dataset

Importing libraries and exploring the dataset

2

Data Cleaning

Handle missing values and inconsistencies.

3

EDA

Visualize data and identify patterns.

4

Feature Engineering

Create new features

5

Key Insights

Analysis reveals several key insights.

Import Required Libraries

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

Exploring the dataset

```
# Load the dataset
df = pd.read_csv('netflix1.csv')
```

Exploring the Data

```
# Display the first few rows of the dataset
df.head()
```

	show_id	type	title	director	country	date_added	release_year	rating	duration	listed_in
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	United States	9/25/2021	2020	PG-13	90 min	Documentaries
1	s3	TV Show	Ganglands	Julien Leclercq	France	9/24/2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...
2	s6	TV Show	Midnight Mass	Mike Flanagan	United States	9/24/2021	2021	TV-MA	1 Season	TV Dramas, TV Horror, TV Mysteries
3	s14	Movie	Confessions of an Invisible Girl	Bruno Garotti	Brazil	9/22/2021	2021	TV-PG	91 min	Children & Family Movies, Comedies
4	s8	Movie	Sankofa	Haile Gerima	United States	9/24/2021	1993	TV-MA	125 min	Dramas, Independent Movies, International Movies

```
df.shape
```

```
(8790, 10)
```


```
# Get concise summary information about the DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8790 entries, 0 to 8789
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8790 non-null   object
 1   type          8790 non-null   object
 2   title         8790 non-null   object
 3   director      8790 non-null   object
 4   country       8790 non-null   object
 5   date_added    8790 non-null   object
 6   release_year  8790 non-null   int64 
 7   rating        8790 non-null   object
 8   duration      8790 non-null   object
 9   listed_in     8790 non-null   object
dtypes: int64(1), object(9)
memory usage: 686.8+ KB
```

Data Cleaning

```
# Check for missing values
df.isnull().sum()
```

```
show_id         0
type            0
title           0
director        0
country         0
date_added      0
release_year    0
rating          0
duration        0
listed_in       0
dtype: int64
```

```
# Converting the date-added(object) to date-time datatype
df['date_added']= pd.to_datetime(df['date_added'])
```

```
# Checking duplicates if any
df.duplicated().sum()
```

```
0
```

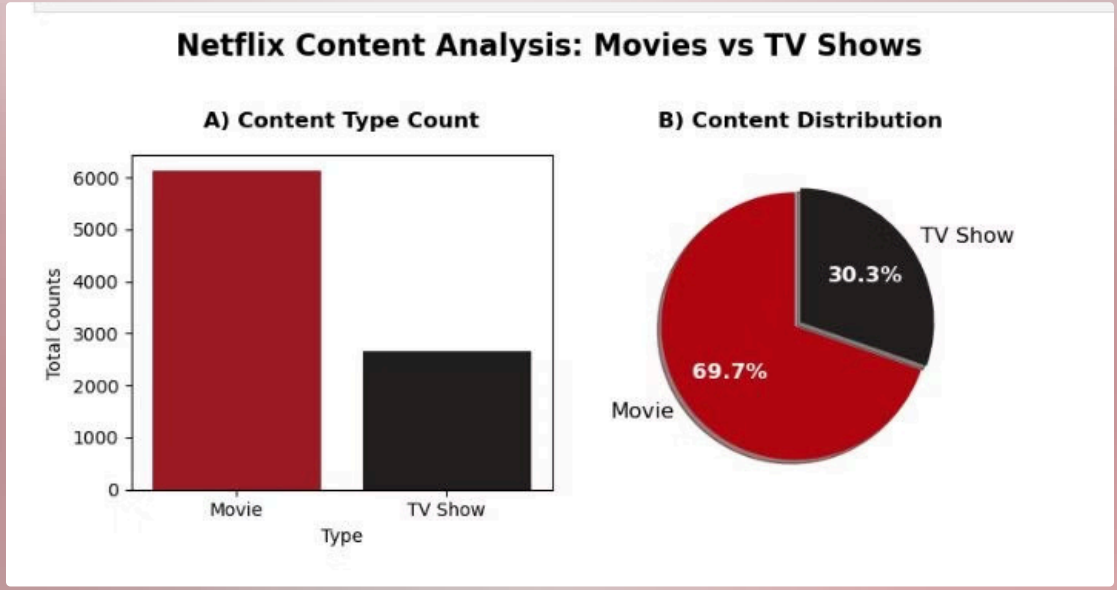
```
#  For numerical description
df.describe()
```

	date_added	release_year
count	8790	8790.000000
mean	2019-05-17 21:44:01.638225408	2014.183163
min	2008-01-01 00:00:00	1925.000000
25%	2018-04-06 00:00:00	2013.000000
50%	2019-07-03 00:00:00	2017.000000
75%	2020-08-19 18:00:00	2019.000000
max	2021-09-25 00:00:00	2021.000000
std	NaN	8.825466

Exploratory Data Analysis (EDA)

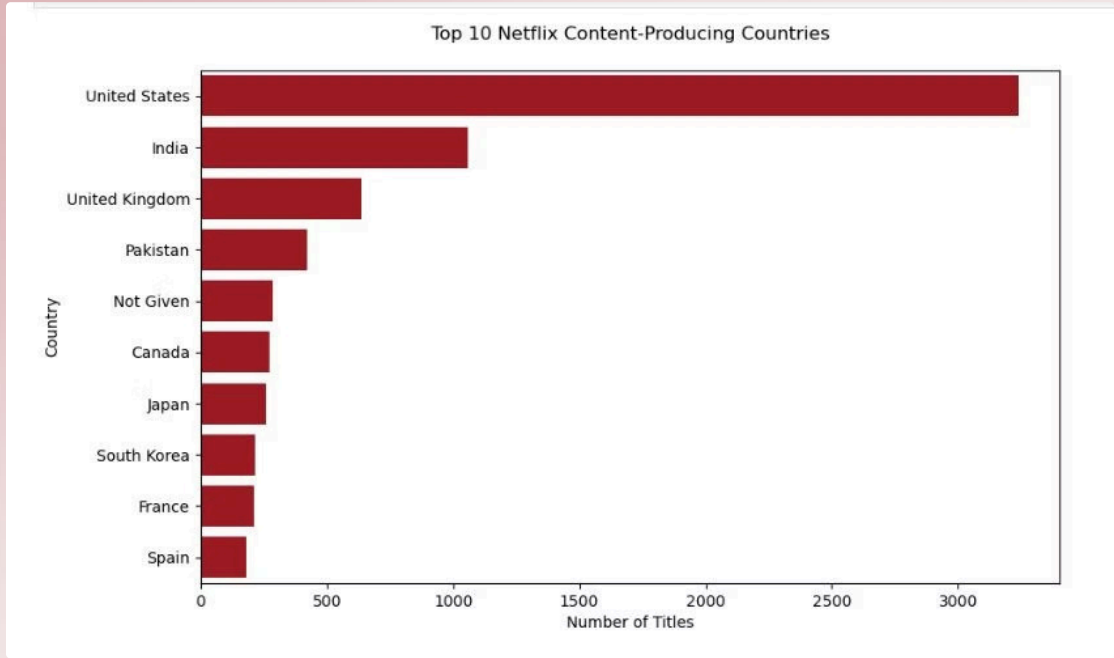
EDA is crucial for uncovering trends in the data. Visualizations help identify key patterns. This provides insights into content performance and user behavior.

- (A). Content Distribution Analysis
- (B). Geographic Analysis
- (C). Director Analysis
- (D). Rating Analysis
- (E). Genre Analysis
- (G). Time Series Analysis



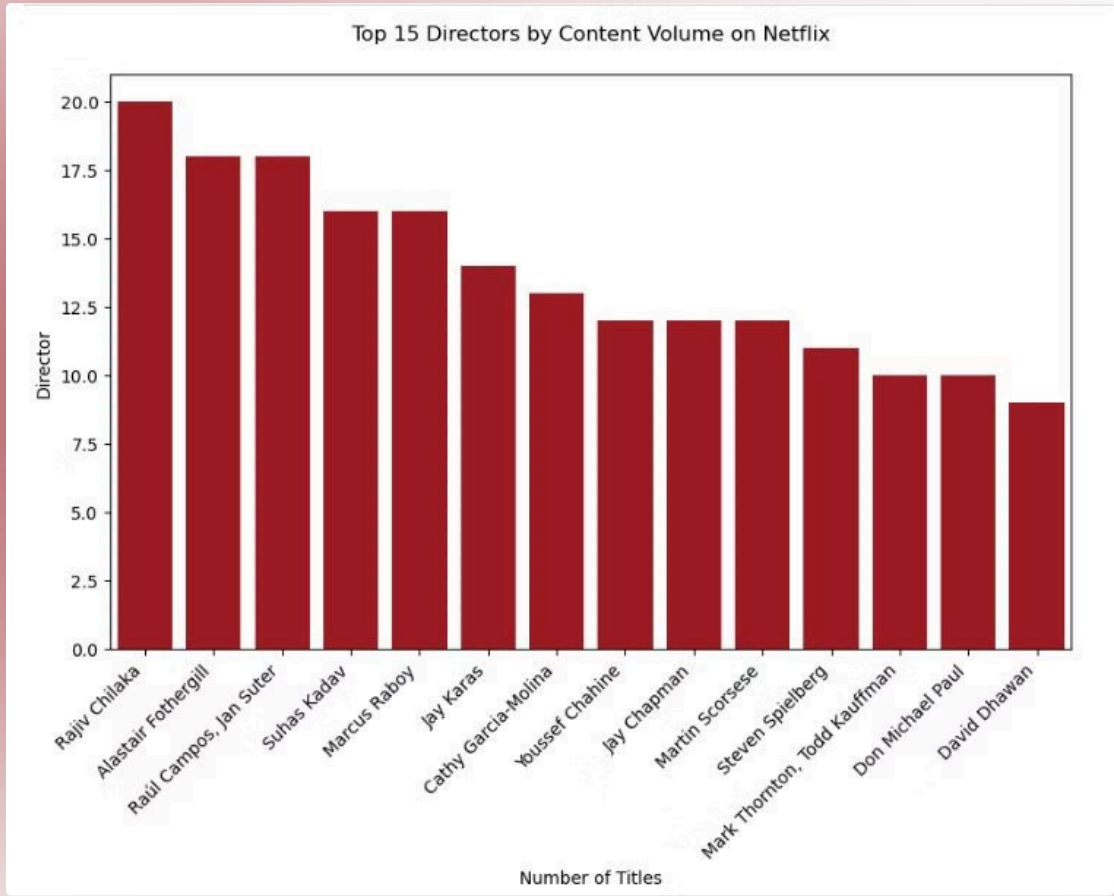
Content Type Distribution: Movies vs TV Shows

```python
# Set up the figure
plt.figure(figsize=(8, 4))  # Wider figure for side-by-side plots

# --- SUBPLOT 1: Count Plot ---
plt.subplot(1, 2, 1)  # 1 row, 2 columns, position 1
ax1 = sns.countplot(x='type', data=df, palette=[NETFLIX_RED, NETFLIX_DARK])  # Netflix colors

ax1.set_title('A) Content Type Count', fontweight='bold', pad=15)
ax1.set_xlabel('Type')
ax1.set_ylabel('Total Counts')

# --- SUBPLOT 2: Pie Chart ---
plt.subplot(1, 2, 2)  # 1 row, 2 columns, position 2
type_counts = df['type'].value_counts()

# Create pie with Netflix colors
patches, texts, autotexts = plt.pie(
    type_counts,
    labels=type_counts.index,
    autopct='%1.1f%%',
    startangle=90,
    colors=[NETFLIX_RED, NETFLIX_DARK],
    explode=(0.05, 0),
    shadow=True,
    textprops={'fontsize': 12}
)

# Style autopct labels
for autotext in autotexts:
    autotext.set_color('white')
    autotext.set_fontweight('bold')

plt.title('B) Content Distribution', fontweight='bold', pad=15)

# --- Final Touches ---
plt.suptitle('Netflix Content Analysis: Movies vs TV Shows',
             fontsize=16, fontweight='bold', y=1.02)
plt.tight_layout()  # Prevent overlapping
plt.show()
```



Top Content-Producing Countries

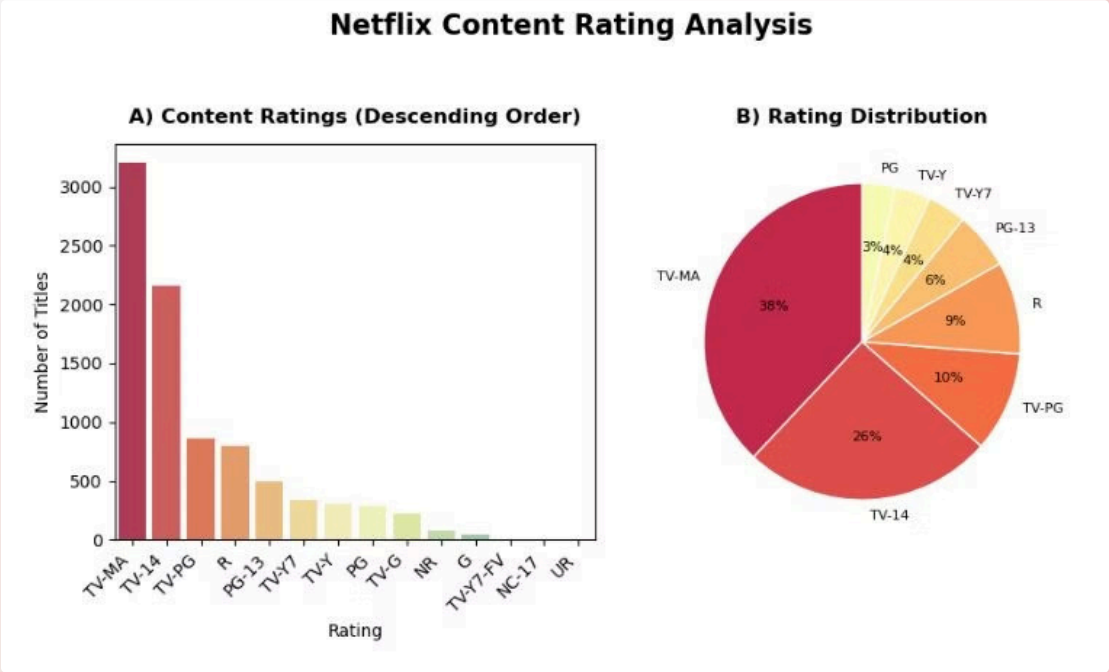
```
# Top 10 Countries by Netflix Content Production

top_countries = df['country'].value_counts().head(10)

# Plot
plt.figure(figsize=(10, 6))
ax = sns.barplot(
    x=top_countries.values,
    y=top_countries.index,
    palette=[NETFLIX_RED]*10 # Gradient red
)
plt.title('Top 10 Netflix Content-Producing Countries',pad=20)
plt.xlabel('Number of Titles')
plt.ylabel('Country')
plt.show()
```



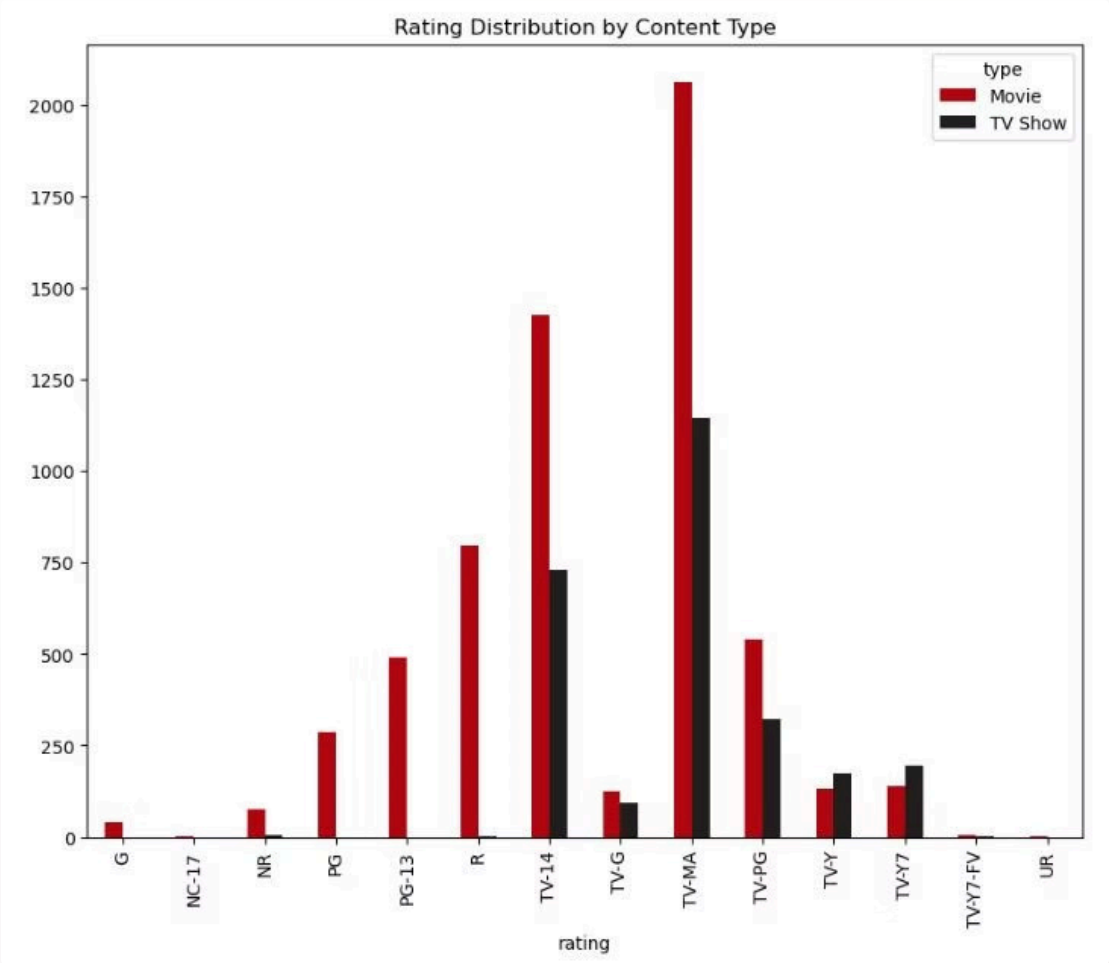
Top most prolific directors

```python
# Top 15 Directors with most numbers of Content
top_directors =
df['director'].value_counts().reset_index().sort_values(by='count', ascending=False)[1:15]


# Plot
plt.figure(figsize=(10, 6))
ax = sns.barplot(
    x=top_directors['director'],
    y=top_directors['count'],
    palette=[NETFLIX_RED]*10
)
plt.title('Top 15 Directors by Content Volume on Netflix',pad=20)
plt.xlabel('Number of Titles')
plt.ylabel('Director')
plt.xticks(rotation=45, ha='right')

plt.show()
```

Top Rating distribution

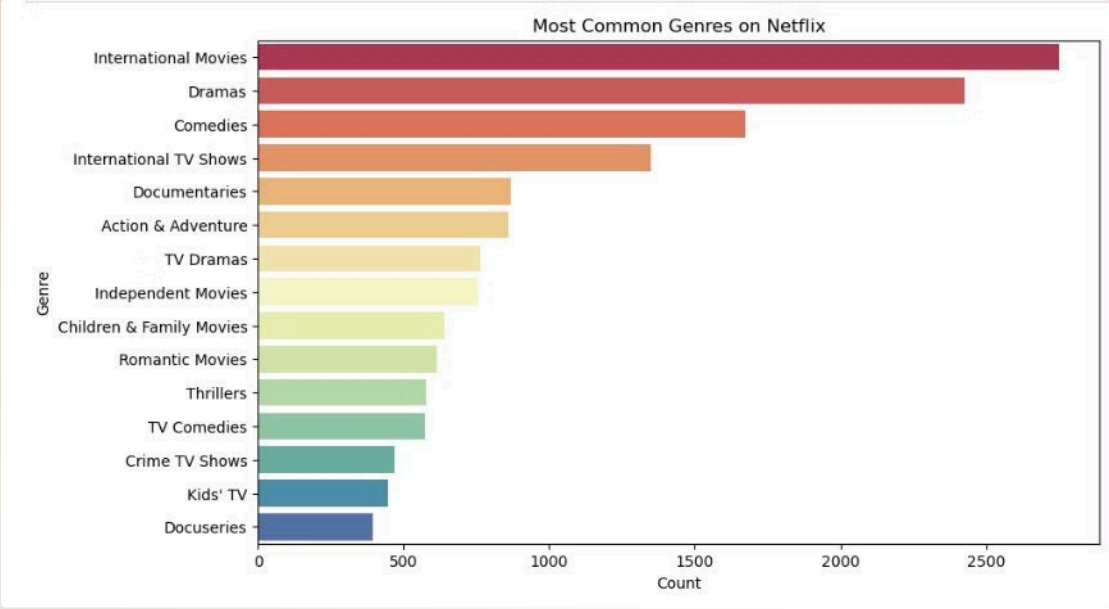



```python
# Set up the figure
plt.figure(figsize=(9, 5))

# --- Data Preparation ---
rating_counts = df['rating'].value_counts().sort_values(ascending=False)
palette = sns.color_palette("Spectral", len(rating_counts))  # Color gradient

# --- SUBPLOT 1: Bar Chart (Descending Order) ---
plt.subplot(1, 2, 1)
ax1 = sns.barplot(x=rating_counts.index, y=rating_counts.values,palette=palette, order=rating_counts.index)

plt.title('A) Content Ratings (Descending Order)', fontweight='bold', pad=12)
plt.xlabel('Rating')
plt.ylabel('Number of Titles')
plt.xticks(rotation=45, ha='right')  # Rotate x-labels for readability

# --- SUBPLOT 2: Pie Chart ---
plt.subplot(1, 2, 2)

ax2 = plt.pie(
    rating_counts[:8],
    labels=rating_counts[:8].index,
    autopct='%.0f%%',
    startangle=90,
    colors=palette,
    wedgeprops={'linewidth': 1, 'edgecolor': 'white'},
    textprops={'fontsize': 8}
)
plt.title('B) Rating Distribution', fontweight='bold', pad=12)

# --- Final Adjustments ---
plt.suptitle('Netflix Content Rating Analysis', fontsize=16, fontweight='bold', y=1.05)
plt.tight_layout()
plt.show()
```



Rating Distribution by Content Type (Movies vs TV Shows)

```
# Rating distribution by type

pd.crosstab(df['rating'], df['type']).plot(kind='bar',
            figsize=(10,8), color = [NETFLIX_RED,NETFLIX_DARK])

plt.title('Rating Distribution by Content Type')
```

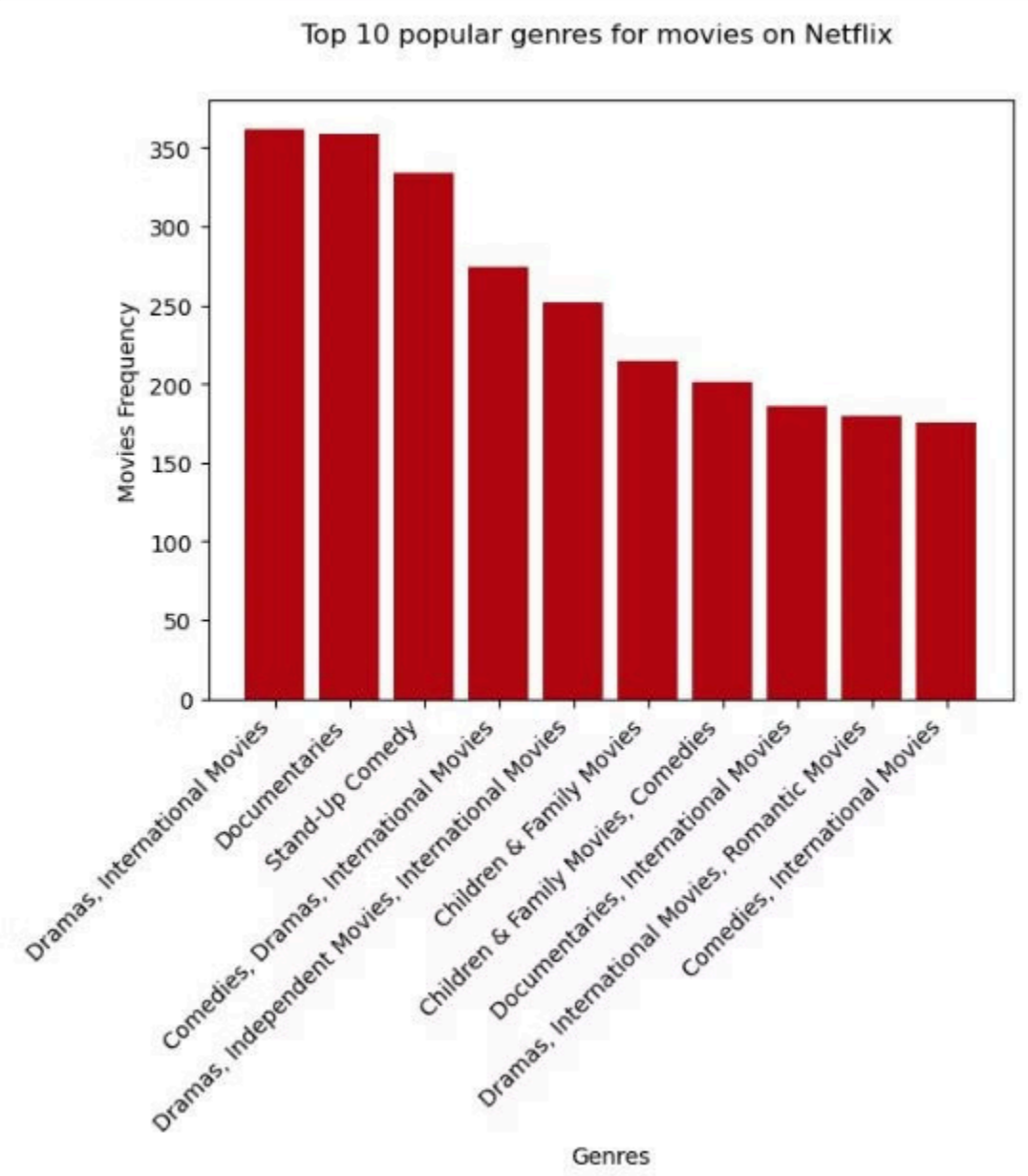


Most Common Genres on Netflix

```
# Split the 'listed_in' column and count genres
df['genres'] = df['listed_in'].apply(lambda x: [genre.strip() for genre in x.split(',')])
all_genres = sum(df['genres'], [])
genre_counts = pd.Series(all_genres).value_counts().head(15)

palette = sns.color_palette("Spectral", len(genre_counts))  # Color gradient

# Plot the most common genres
plt.figure(figsize=(10, 6))
sns.barplot(x=genre_counts.values, y=genre_counts.index,
palette=palette )
plt.title('Most Common Genres on Netflix')
plt.xlabel('Count')
plt.ylabel('Genre')
plt.show()
```

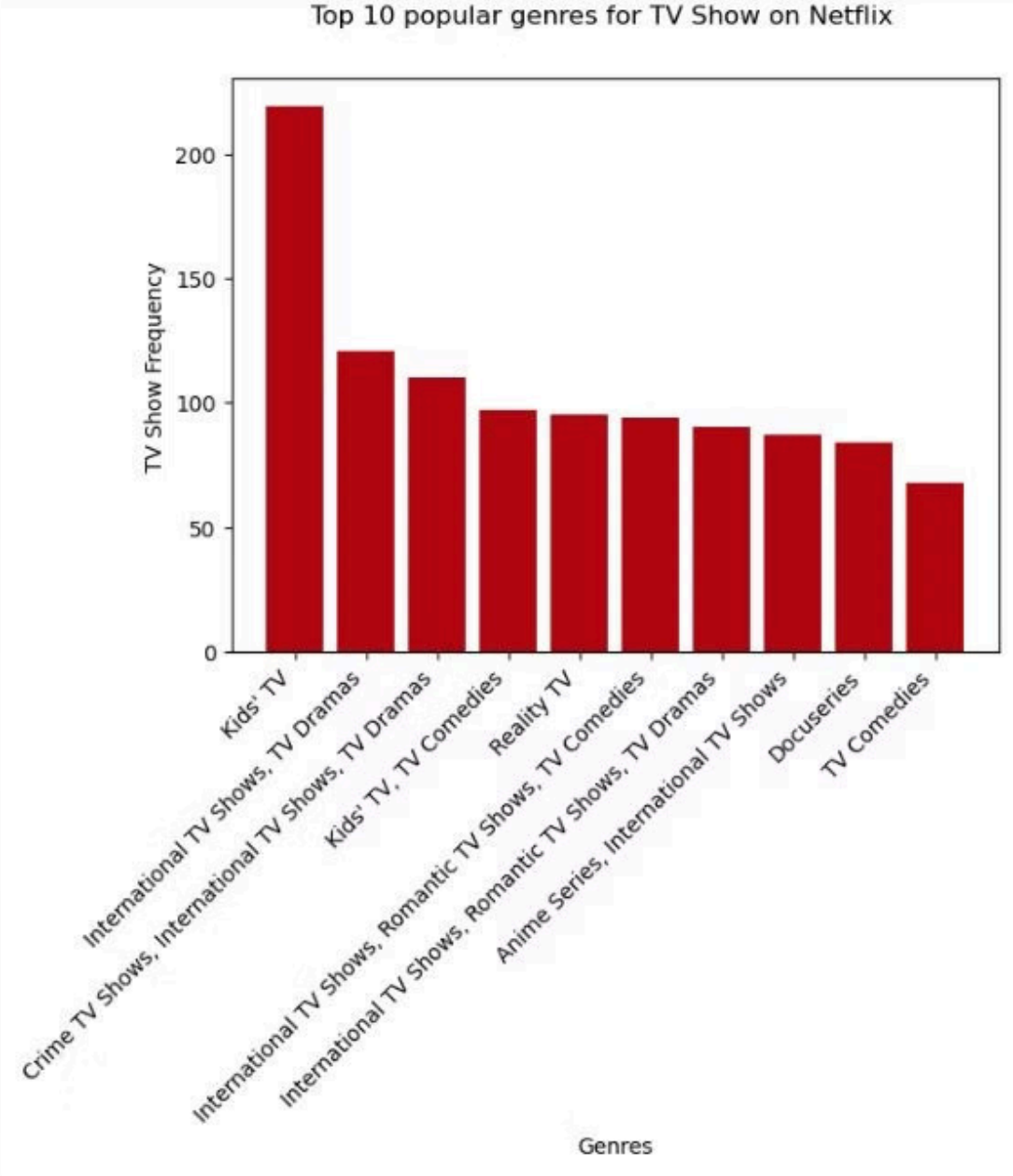
Tree Map of Genre Distribution

```
genre_count_all = pd.Series(all_genres).value_counts().reset_index()
genre_count_all.columns = ['Genre', 'Counts']

plt.figure(figsize=(15, 10))
fig = px.treemap(genre_count_all, path=['Genre'], values='Counts'
                 , title='All Genre Distribution', width=1200,height=700)
fig.show()
```



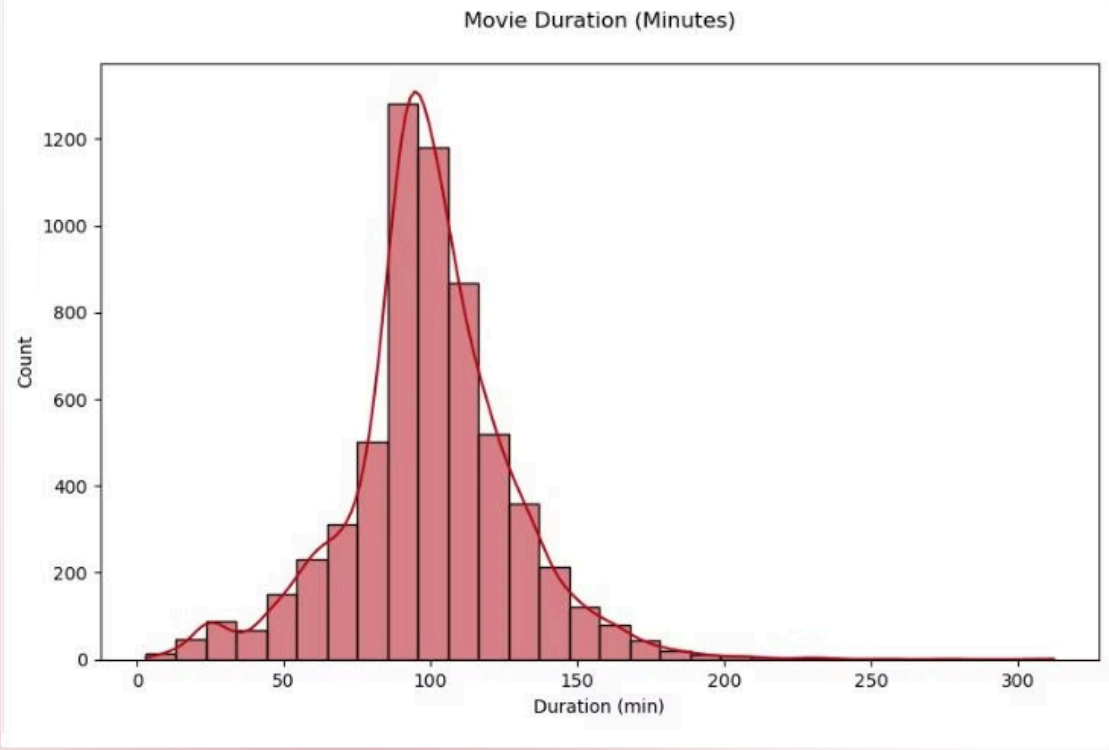
Most popular movie genres

```
# Top 10 popular movie genres

popular_movie_genre=df[df['type']==
'Movie'].groupby("listed_in").size().sort_values(ascending=False)[:10]

plt.bar(popular_movie_genre.index, popular_movie_genre.values, color = NETFLIX_RED)
plt.xticks(rotation=45, ha='right')
plt.xlabel("Genres")
plt.ylabel("Movies Frequency")
plt.suptitle("Top 10 popular genres for movies on Netflix")
plt.show()
```



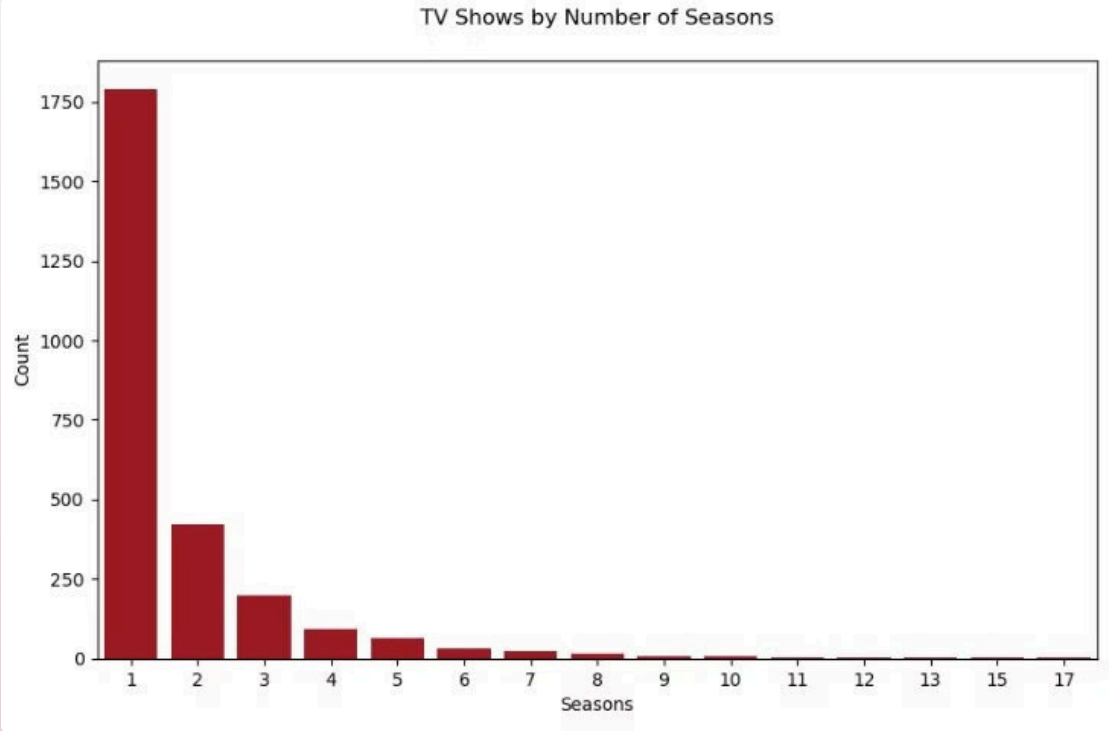
Most popular TV Shows genres

```
# Top 10 popular TV Shows genres

popular_series_genre=df[df['type']==
'TV Show'].groupby("listed_in").size().sort_values(ascending=False)[:10]

plt.bar(popular_series_genre.index, popular_series_genre.values, color = NETFLIX_RED)
plt.xticks(rotation=45, ha='right')
plt.xlabel("Genres")
plt.ylabel("TV Show Frequency")
plt.suptitle("Top 10 popular genres for TV Show on Netflix")
plt.show()
```



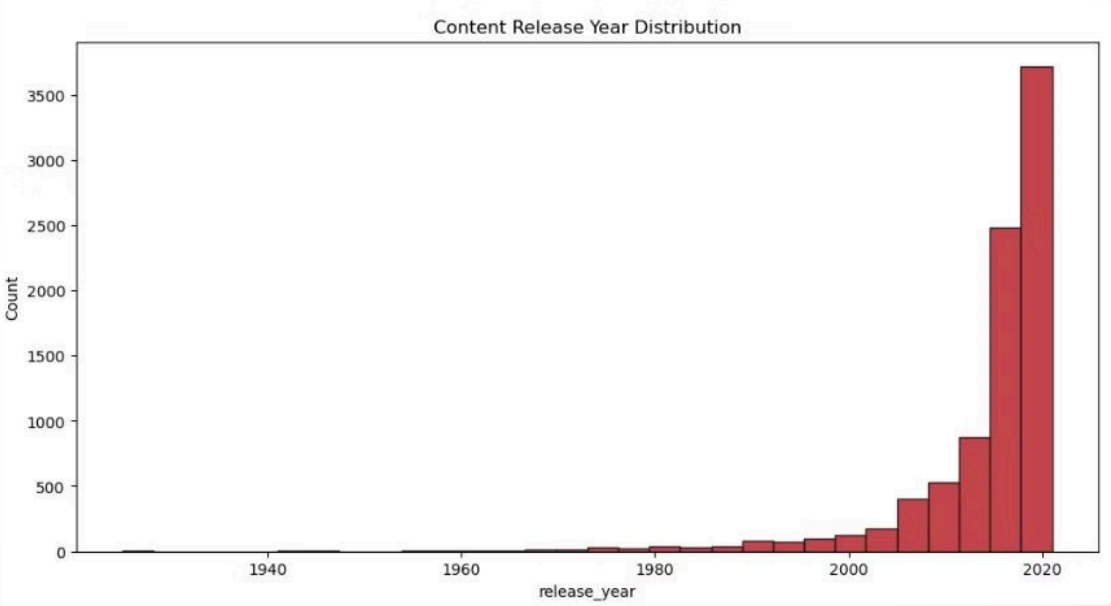
Movie duration distribution

```python
# Movie duration distribution

# Extract minutes for movies
movies = df[df['type'] == 'Movie']
movies['duration_min'] = movies['duration'].str.extract('(\d+)').astype(float)

# Plot
plt.figure(figsize=(10, 6))
ax = sns.histplot(
    movies['duration_min'],
    bins=30,
    color=NETFLIX_RED,
    edgecolor='black',
    kde=True
)
plt.title('Movie Duration (Minutes)', pad=20)
plt.xlabel('Duration (min)')
plt.ylabel('Count')
plt.show()
```



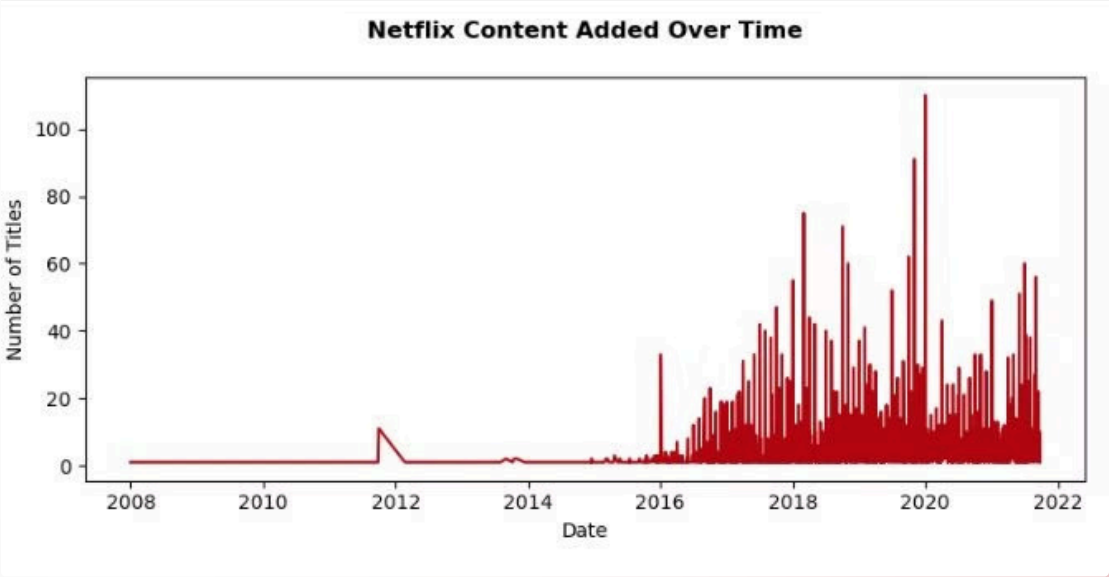
TV show seasons distribution

```
# TV show seasons distribution

# Extract seasons for TV shows
tv_shows = df[df['type'] == 'TV Show']
tv_shows['seasons'] = tv_shows['duration'].str.extract('(\d+)').astype(int)
seasons_counts = tv_shows['seasons'].value_counts().sort_index()

# Plot
plt.figure(figsize=(10, 6))
ax = sns.barplot(
    x=seasons_counts.index,
    y=seasons_counts.values,
    palette=[NETFLIX_RED]*10
)
plt.title('TV Shows by Number of Seasons', pad=20)
plt.xlabel('Seasons')
plt.ylabel('Count')
plt.show()
```



Content Release Year Distribution

```
#  content release by years
plt.figure(figsize=(12,6))
sns.histplot(df['release_year'], bins=30, color=NETFLIX_RED ,edgecolor=NETFLIX_DARK)
plt.title('Content Release Year Distribution')
```



Content Addition on Netflix over time

```python
# 1. Group data
ts_data = df.groupby('date_added')['show_id'].count().reset_index(name='count')

# 2. Create the plot with Seaborn
plt.figure(figsize=(8,4))
ax = sns.lineplot(
    data=ts_data,
    x='date_added',
    y='count',
    color=NETFLIX_RED,
    linewidth=1.5,
)
plt.title('Netflix Content Added Over Time', fontweight='bold',pad=20)
plt.xlabel('Date')
plt.ylabel('Number of Titles')

plt.tight_layout()  # Prevent label clipping
plt.show()
```

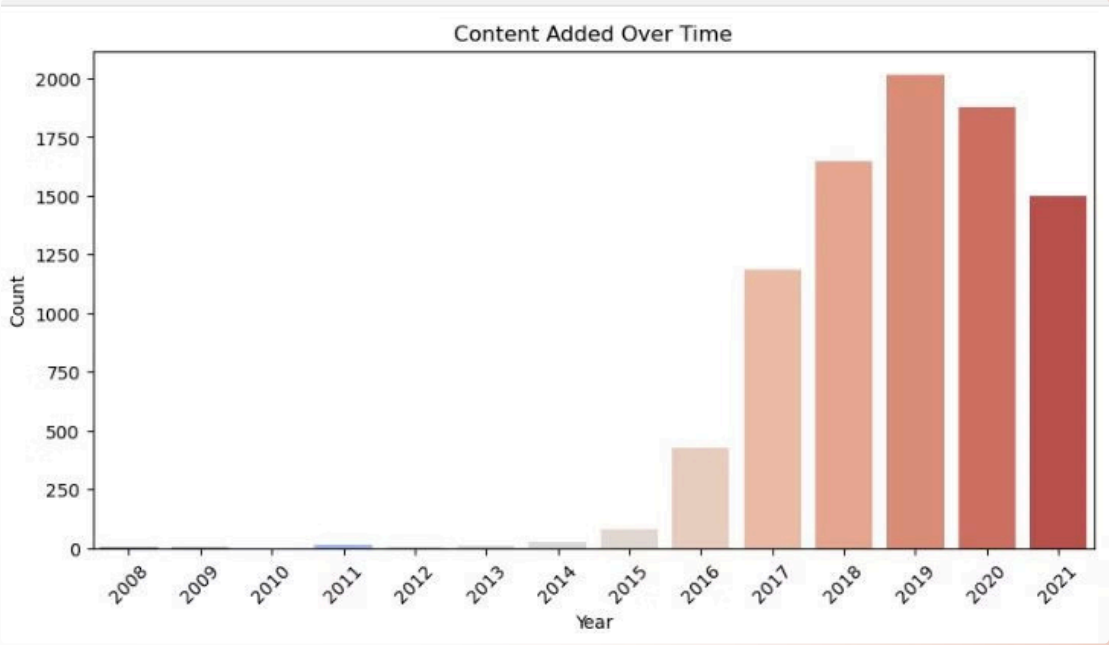
Feature Engineering

Feature Engineering focuses on extracting maximum value from each column while maintaining interpretability.

Creating new columns for deeper analysis and suitable visualization. Here some necessary feature engineering require for further visualization.

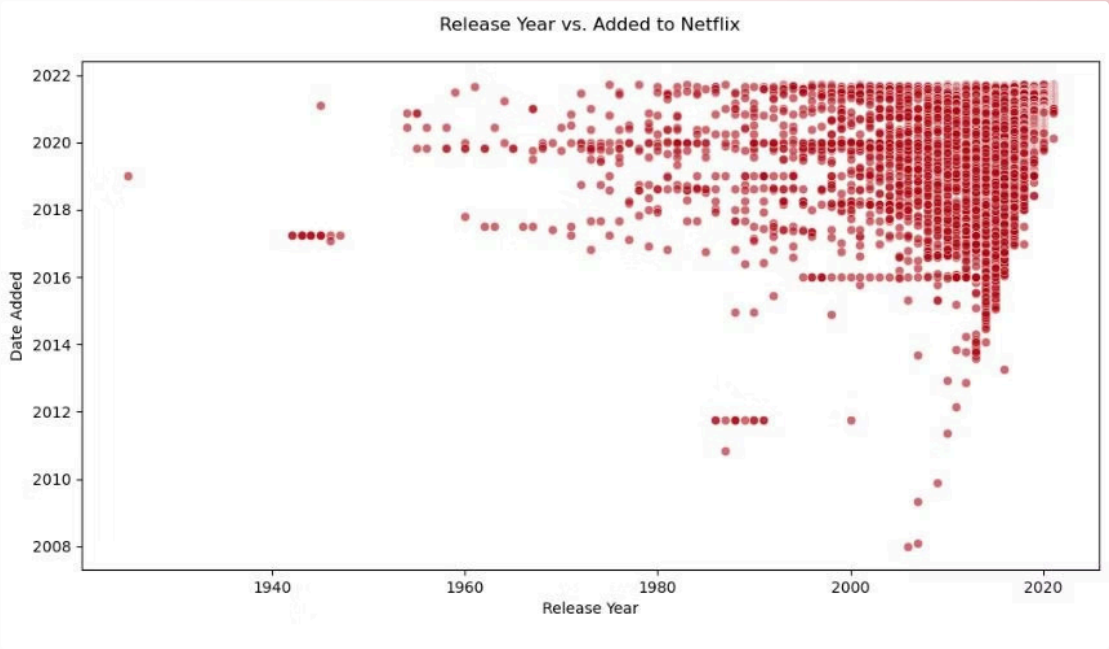
Created Three new columns 'year_added', 'month_added', 'day_added' from an existing column name 'date_added'.

```
#  Creating 3 New Columns

df['year_added']=df['date_added'].dt.year
df['month_added']=df['date_added'].dt.month
df['day_added']=df['date_added'].dt.day
```



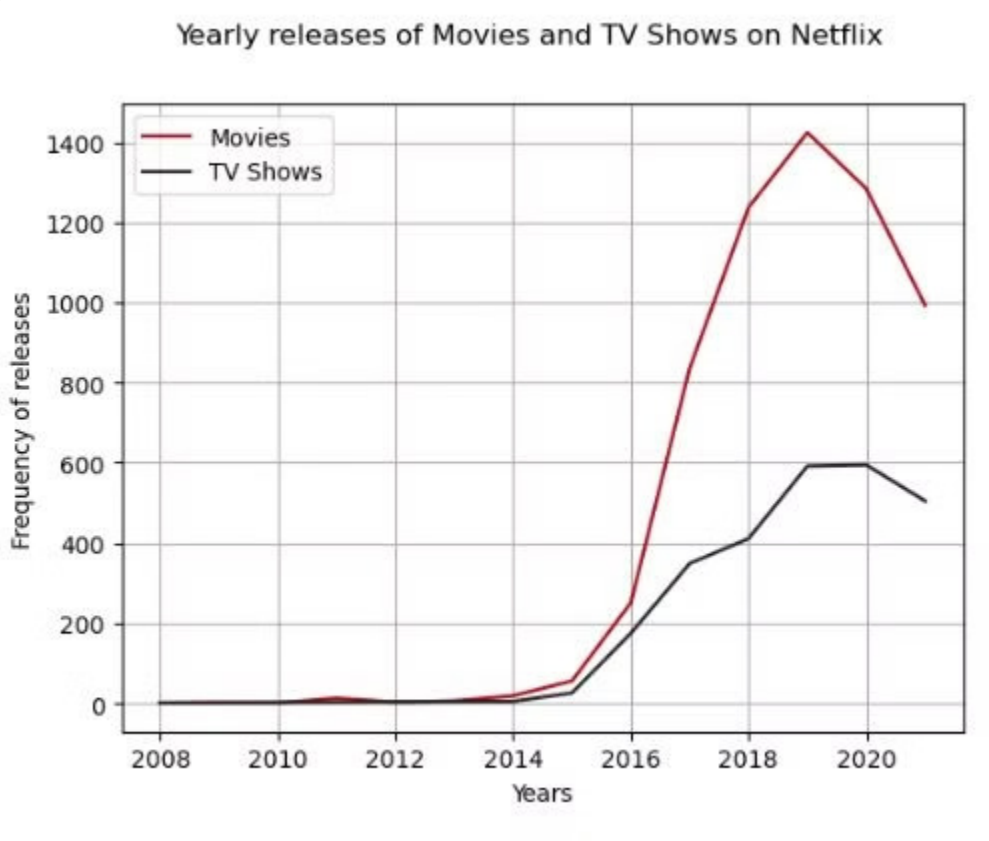
Content Addition Distribution by Years on Netflix

```
# Using the new created columns from Feature Engineering

# Plot content added over the years
plt.figure(figsize=(10, 5))
sns.countplot(x='year_added', data=df, palette='coolwarm')
plt.title('Content Added Over Time')
plt.xlabel('Year')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```



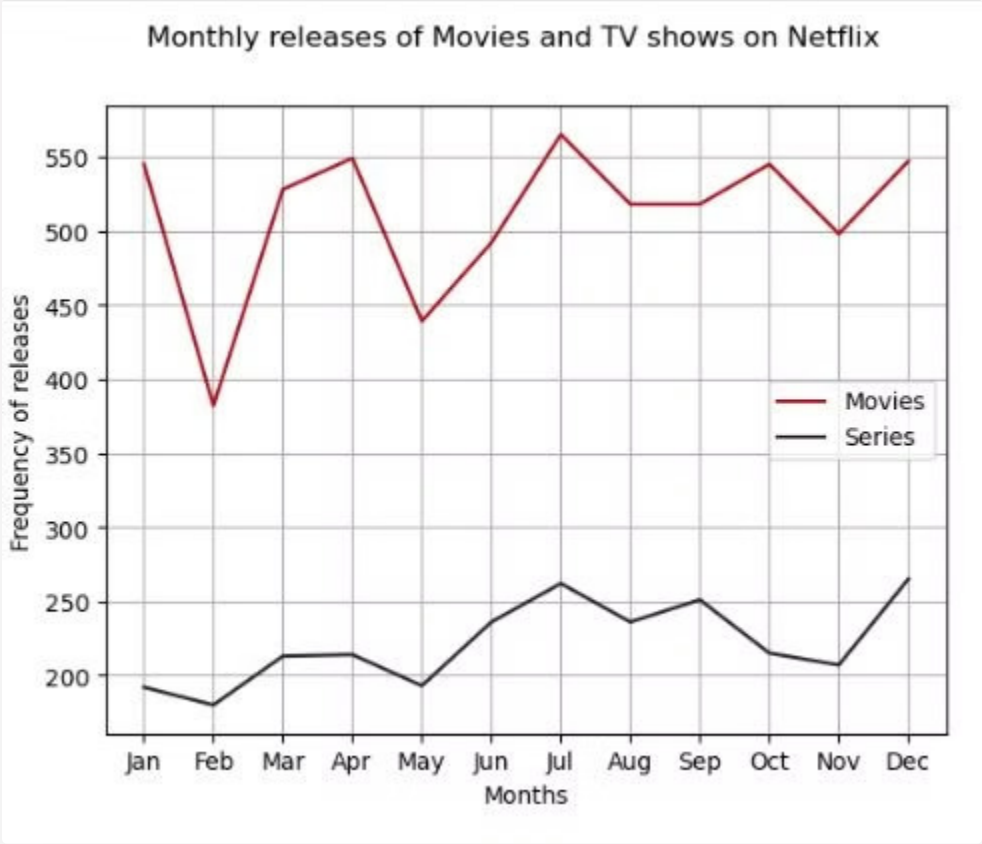
Release Year vs. Added to Netflix

```
#  Plot
plt.figure(figsize=(12, 6))
ax = sns.scatterplot(
    data=df,
    x='release_year',
    y='date_added',
    color=NETFLIX_RED,
    alpha=0.6  # Transparency for overlapping points
)
plt.title('Release Year vs. Added to Netflix', pad=20)
plt.xlabel('Release Year')
plt.ylabel('Date Added')
plt.show()
```



Yearly releases of Movies and TV Shows on Netflix

```
yearly_movie_releases=df[df['type']=='Movie']['year_added'].value_counts().sort_index()
yearly_series_releases=df[df['type']=='TV Show']['year_added'].value_counts().sort_index()

plt.plot(yearly_movie_releases.index,yearly_movie_releases.values, label='Movies', color = NETFLIX_RED)
plt.plot(yearly_series_releases.index,yearly_series_releases.values, label='TV Shows', color = NETFLIX_DARK)
plt.xlabel("Years")
plt.ylabel("Frequency of releases")
plt.grid(True)
plt.suptitle("Yearly releases of Movies and TV Shows on Netflix")
plt.legend()
```



Monthly releases of Movies and TV shows on Netflix

```
monthly_movie_release=df[df['type']=='Movie']['month_added'].value_counts().sort_index()
monthly_series_release=df[df['type']=='TV Show']['month_added'].value_counts().sort_index()

plt.plot(monthly_movie_release.index,monthly_movie_release.values, label='Movies', color = NETFLIX_RED)
plt.plot(monthly_series_release.index,monthly_series_release.values, label='Series', color = NETFLIX_DARK)
plt.xlabel("Months")
plt.ylabel("Frequency of releases")
plt.xticks(range(1, 13), ['Jan', 'Feb', 'Mar', 'Apr', 'May','Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])
plt.legend()
plt.grid(True)
plt.suptitle("Monthly releases of Movies and TV shows on Netflix")
plt.show()
```

Key Insights & Actionable Recommendations

Our analysis reveals several key insights. Each insight leads to actionable recommendations. These recommendations can improve content strategy and user engagement.

1. Movies dominate the catalog, making up ~70% of the total content, while TV shows account for only ~30%.
2. Top Directors are Rajiv Chilaka (20 titles), Alastair Fothergill (18 titles), Raúl Campos & Jan Suter (18 titles), Suhas Kadav (16 titles), Marcus Raboy (16 titles).
3. TV-MA, TV-14 and R-rated content make up ~70% of the catalog, setting Netflix apart from family-focused.
4. Movies skew toward adult audiences (R, PG-13), while TV balances broader demographics(TV-Y, TV-Y7,TV-14) and No G/PG-rated TV shows exist.
5. International Movies/TV Shows dominate the library(~30% of listed genre), Dramas (2,426) and TV Dramas (762) are highlight, Comedies (1,674) and TV Comedies (573) indicate strong demand
6. Dramas, International Movies" (362), Documentaries (359), Stand-Up Comedy (334) are the top hybrid categories of Movies Genre. And "Kids' TV" (219), "International TV Shows, TV Dramas" (121) and "Crime TV Shows, International TV Shows, TV Dramas" (110)are Top of TV Shoes Genres
7. Movie Duration have a strong preference for 90–120-minute films and few movies exceed 150 minutes or fall below 60 minutes. And TV Show Seasons Distribution shows a steep decline after Season 3, with most TV shows (1,000+) having ≤3 seasons. .

Thank You

by Nimit Tiwari