Netflix Data Analysis

Project Report

By :- Nikita Vishwakarma

Date:- 30-01-25

Table Of Content:-

1. Introduction
2. Data Preprocessing & Cleaning
3. Exploratory Data Analysis (EDA)
4. Time-Series Analysis
5. Content Categorization
6. Rating Distribution
7. Duration aspect
8. Conclusion

Introduction:-

Netflix is one of the largest streaming platforms, offering a vast collection of movies, TV shows, and documentaries across various genres and languages. The primary objective of this project is to analyze Netflix's dataset to uncover patterns, trends, and insights related to its content library.

Scope of Analysis-

Content Types – Understanding the distribution of Movies vs. TV Shows.
Release Trends – Identifying how content additions have changed over time.
Genre Popularity – Finding the most frequent and popular genres.
Ratings Distribution – Analyzing audience classifications (e.g., TV-MA, PG-13).
Duration Analysis – Examining the length of movies and TV shows**.**
Regional Content Trends – Exploring top content-producing countrie

Data Preprocessing & Cleaning:-

Objective:

To load, inspect, clean, and preprocess the Netflix dataset (`netflix1.csv`) by checking for data type issues, handling missing values, and removing duplicates to ensure a clean dataset ready for further analysis.

Steps:-

1: Load the Dataset

2: Display DataFrame Info

3: Convert "date_added" to Datetime Format

4: Check for Missing Values

5: Drop Duplicate Values

Code:-

```
df = pd.read_csv('netflix1.csv')
df.head()

# Display summary info about the DataFrame
df.info()

# Convert the column to datetime
print("Convert Coloumn to Datetime")
df["date_added"] = pd.to_datetime(df["date_added"])

#check null valus in dataset
print("Check NUll Values in Dataset")
df.isnull().sum()

#drop duplicate values
print("drop duplicate values")

df.drop_duplicates(inplace=True)
```

Explanation:-

- **Data Inspection & Preprocessing:**
  The code first displays the summary information of the dataset (`df.info()`), converts the `date_added` column to a datetime format for easier manipulation, and checks for missing values using `isnull().sum()`.
- **Data Cleaning:**
  The code removes any duplicate rows in the dataset using `drop_duplicates()` to ensure the data is clean and free of redundancy, ready for analysis.

Output:-

```
<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
Convert Coloumn to Datetime
drop duplicate values
```

```
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
```

Exploratory Data Analysis (EDA):-

This is Data Visualization, which is part of Exploratory Data Analysis (EDA). It helps to understand the data before performing further processing or modeling

Type of Cetegory-

Objective:-

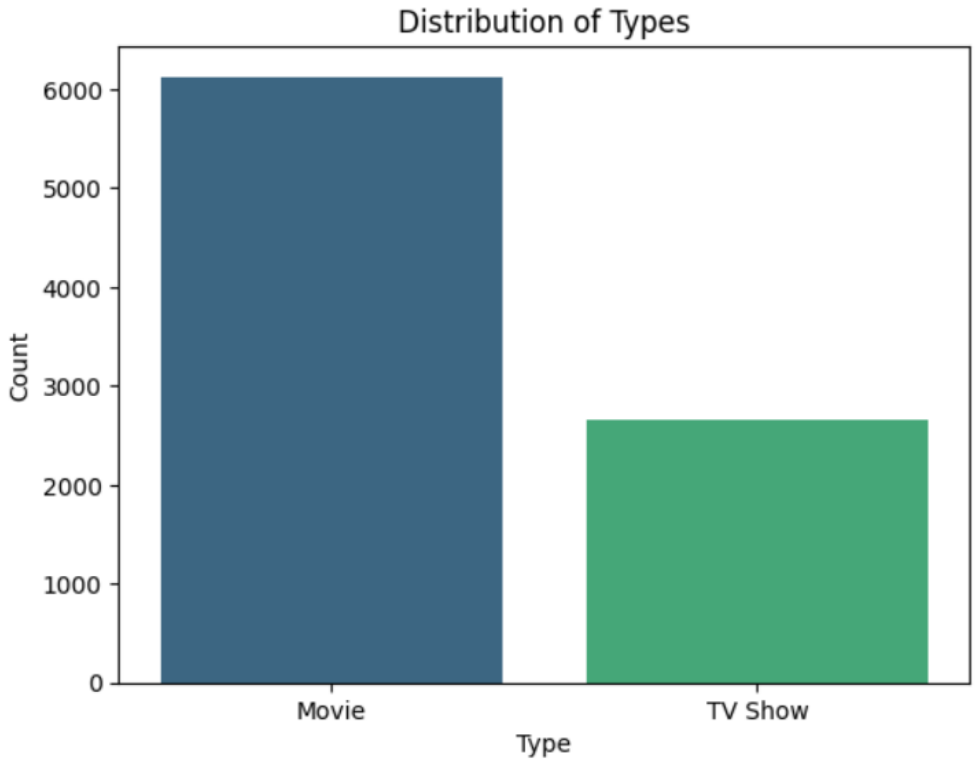
To visualize the distribution of different in the dataset using a bar plot.

Steps:-

1. Create the Bar Plot

Code:-

```
# Create the count plots
sns.barplot(x=type_values.index, y=type_values.values, palette='viridis')
# Set the title and labels
plt.title('Distribution of Types')
plt.xlabel('Type')
plt.ylabel('Count')

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



Explanation:-

- This code visualizes the distribution of different types **(e.g., Movie, TV Show)** in the dataset using a bar plot with Seaborn and Matplotlib

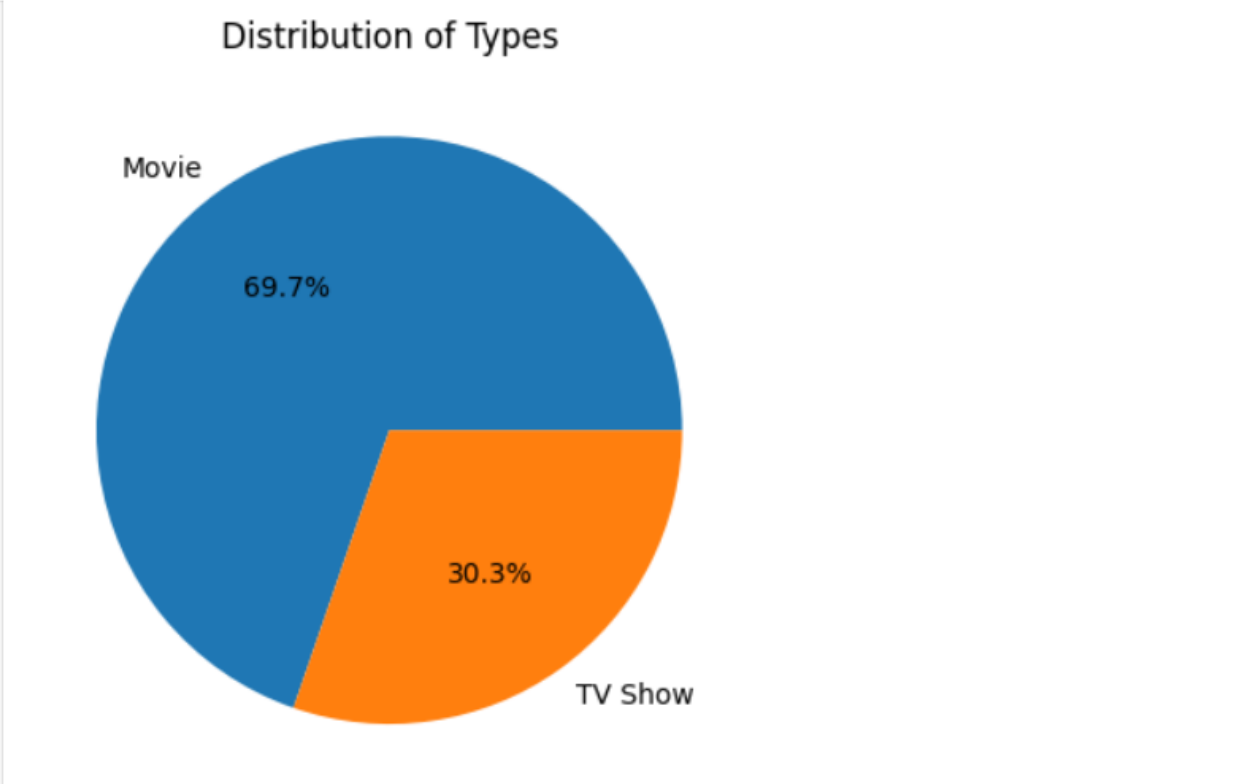
Percentage Distribution in Type Category-

Objective:-

To visualize the proportion of different types (e.g., Movies vs. TV Shows) in the dataset using a pie chart**,** making it easier to compare their distribution at a glance.

Code:-

```
# Create a pie chart
plt.pie(type_values, labels=type_values.index, autopct='%1.1f%%')
Z
# Set the title
plt.title('Distribution of Types')

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



Explanation:-

- This code visualizes the distribution of different types (e.g., Movies, TV Shows) in the dataset using a pie chart with Matplotlib.

Top 10 Most Frequent Categories in "listed_in":-

Objective:-

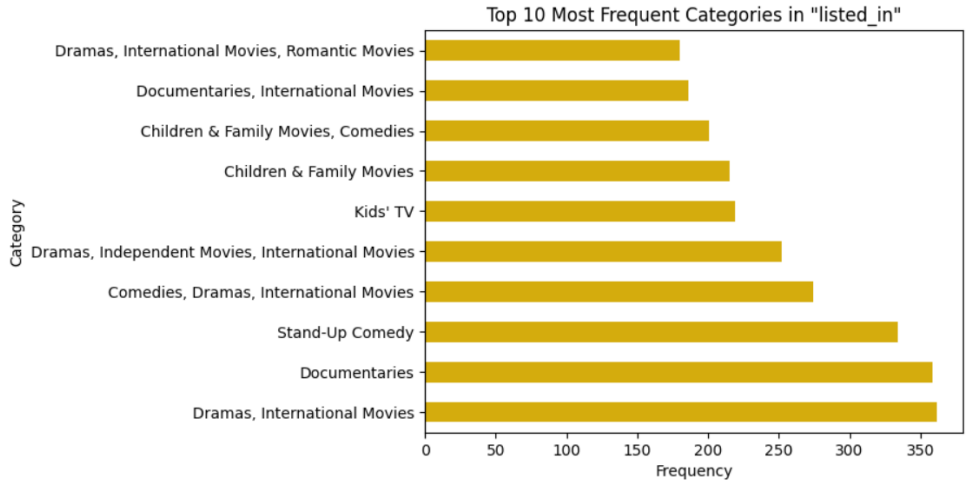
To visualize the top 10 most frequent categories in the "listed_in" column using a horizontal bar chart, helping identify the most common content genres or categories in the dataset.

Step:-

1. Plot the Horizontal Bar Chart:

Code:-

```
# Plot the result
top10_listen_in.plot(kind='barh', color='#d4ac0d')

# Add labels and title
plt.xlabel('Frequency')
plt.ylabel('Category')
plt.title('Top 10 Most Frequent Categories in "listed_in"')

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



Explanation:-

- Understand the proportion of each category (e.g., % of Movies vs. TV Shows). Identify imbalances in the dataset.
- Make the data more interpretable for analysis.

Top 10 Most Prolific Directors-

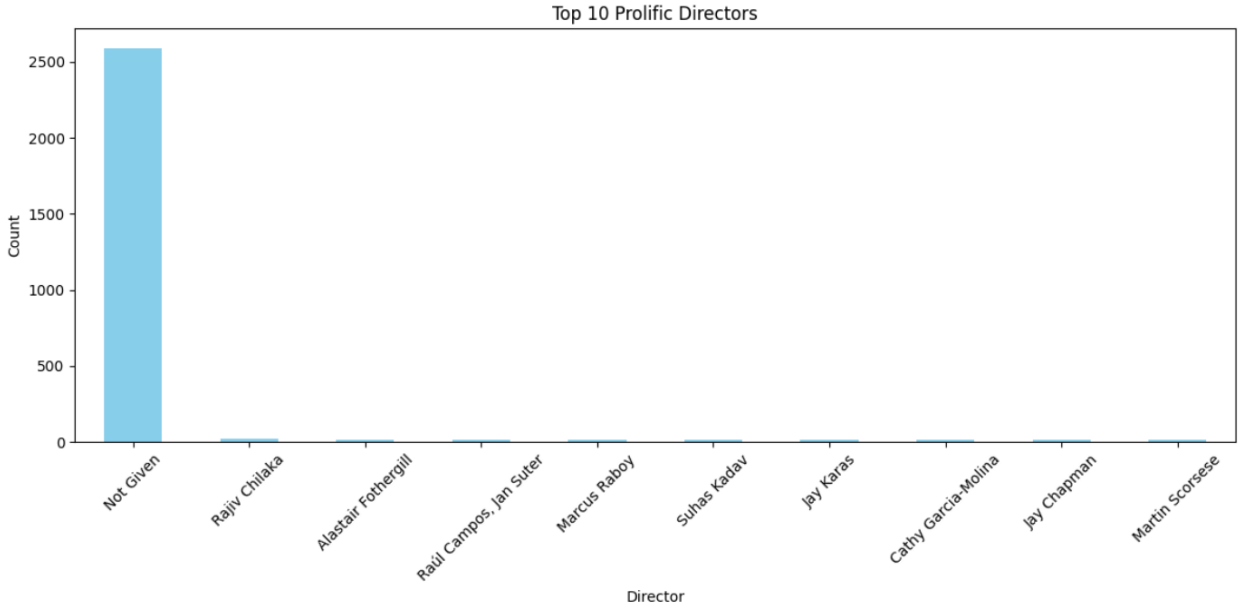
Objective:-

To visualize the top 10 most prolific directors in the dataset using a bar chart**,** helping identify which directors have the most content in the dataset

Step:- 1. Plot the Bar Chart

Code:-

```
# Plot the bar chart
top10_directors.plot(kind='bar', color='skyblue', figsize=(12, 6))
plt.title('Top 10 Prolific Directors')
plt.xlabel('Director')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Explanation:-

- Helps identify the most influential or frequently featured directors in the dataset.Useful for trend analysis in filmmaking (e.g., which directors contribute the most?).
- Supports recommendation systems by understanding director-based content trends.

Top 10 Countries by Content Count-

Objective:-

To visualize the top 10 countries with the most content in the dataset using a bar chart, helping identify which countries produce or feature the most content.

Step:-

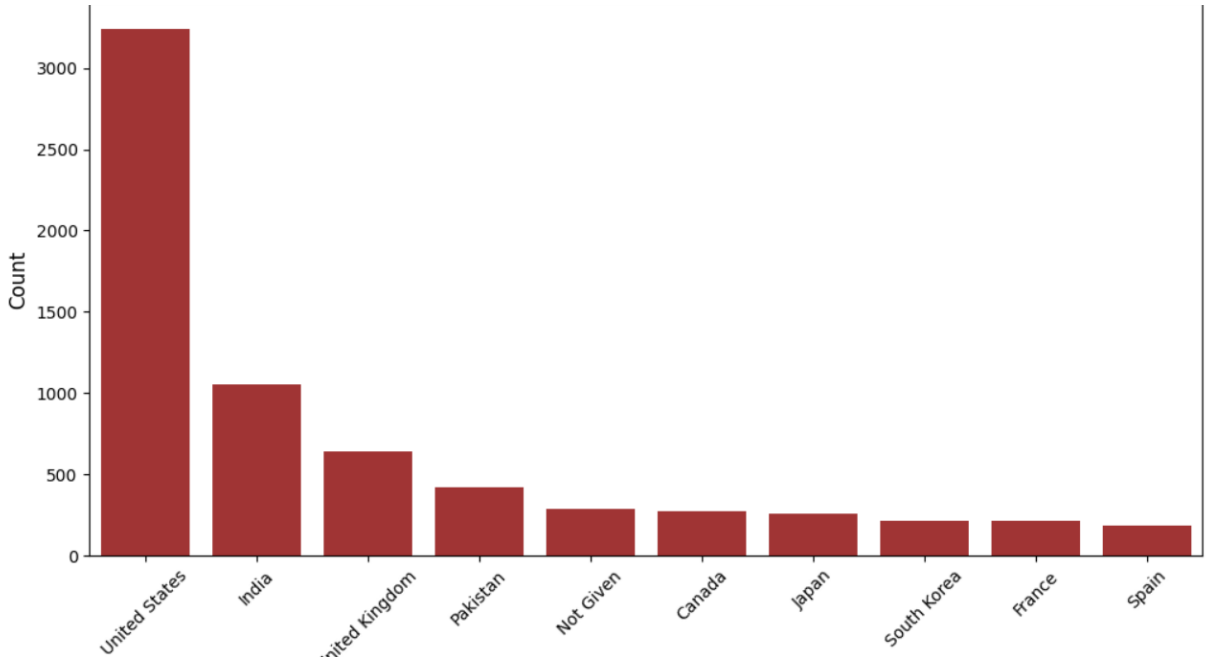
1. Get the Top 10 Countries

2. Convert to DataFrame for Seaborn Compatibility

3. Plot the Data using Seaborn

Code:-

```
# Get the top 10 countries
top10_countries = df['country'].value_counts().head(10)

# Convert to DataFrame for seaborn compatibility
top10_countries_df = top10_countries.reset_index()
top10_countries_df.columns = ['Country', 'Count']

# Plot using seabor
plt.figure(figsize=(12, 6))
sns.barplot(x='Country', y='Count', data=top10_countries_df, color='#B22222')
plt.title('Top 10 Countries by Content Count')
plt.xlabel('Country')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```



Explanation:-

- Helps understand the distribution of content by country**,** showing which countries dominate the da taset. Useful for identifying geographical content trends**.**
- can guide country-specific content recommendation

Time-Series Analysis:-

Trends in Content Types Over the Years (Based on Release Year-

Objective:-

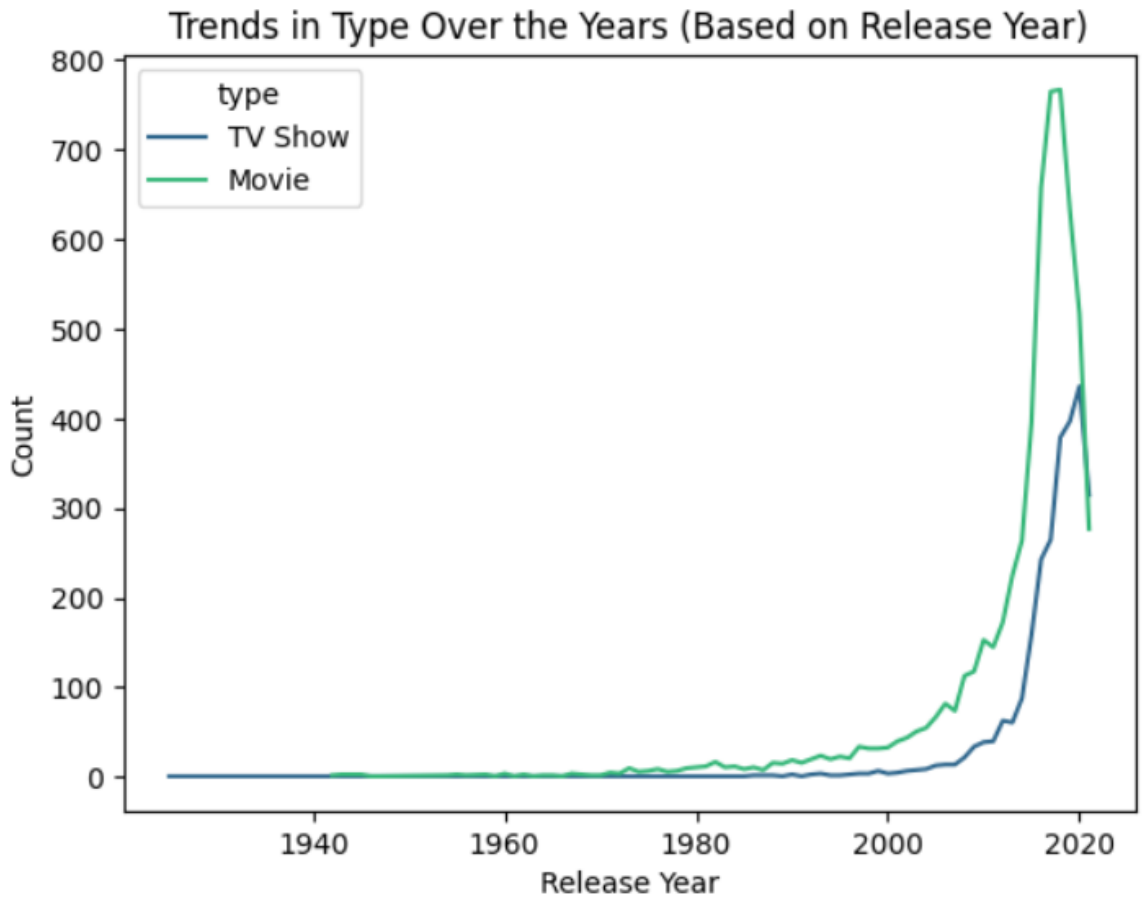
To visualize the trends in content types (e.g., Movies, TV Shows) over the years using a line plot, helping identify how the number of different types has evolved over time based on their release year.

Step:-

1. Group Data by Release Year and Type
2. Create a Line Plot to Visualize the Trends

Code:-

```
# Group by 'Release_year' and 'type', and count occurrences
df_grouped = df.groupby(['release_year', 'type']).size().reset_index(name='count')

# Create a bar plot to visualize the trend
sns.lineplot(x='release_year', y='count', hue='type', data=df_grouped, palette='viridis')

# Set the title and labels
plt.title('Trends in Type Over the Years (Based on Release Year)')
plt.xlabel('Release Year')
plt.ylabel('Count')

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



Explanation:-

- Helps visualize how the number of Movies and TV Shows has changed over the years.
- Provides insights into content production trends, helping identify any significant shifts in the industry.
- Useful for understanding historical content preferences.

Number of Movies/TV Shows Released Per Year-

Objective:-

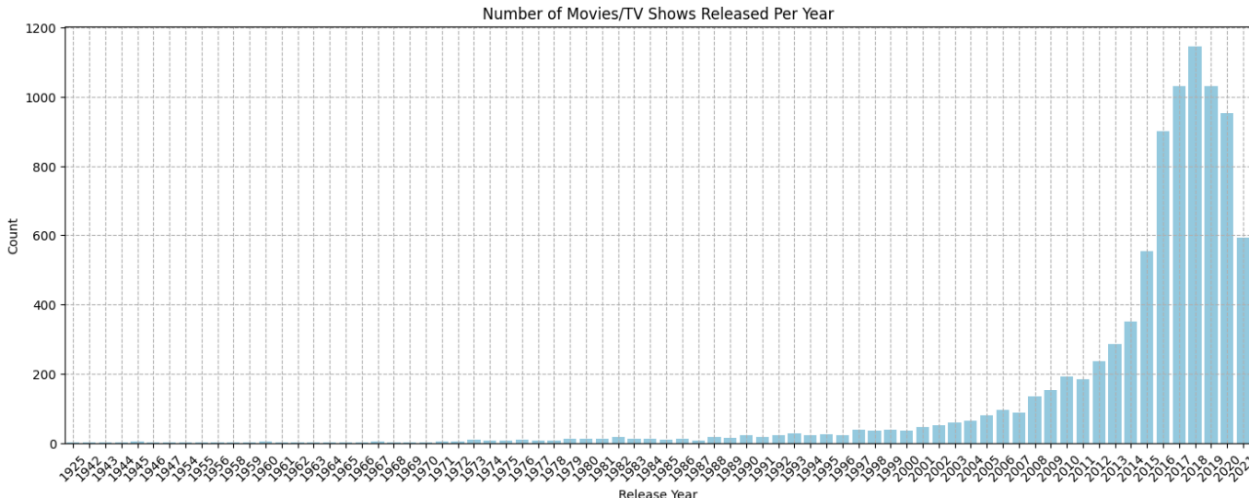
To visualize the number of Movies and TV Shows released each year using a bar chart, helping identify patterns in content releases over time.

Step:-

1. Count the Number of Movies/TV Shows Released Each Year
2. Plot the Data

Code:-

```
# Count the number of movies/TV shows released each year
release_year_counts = df['release_year'].value_counts()# Sort by year

# Plot the data
plt.figure(figsize=(17, 6))  # Set figure size
sns.barplot(x=release_year_counts.index, y=release_year_counts.values, color='skyblue')
plt.title('Number of Movies/TV Shows Released Per Year')
plt.xlabel('Release Year')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(linestyle = '--')
plt.show()
```



Explanation:-

- Helps to understand the annual trends in content releases.
- Provides insights into changes in content production over time.
- Useful for recognizing any peaks or drops in content release frequency

Evolution of Content Release Over Time (Movies vs TV Shows)-

Objective:-

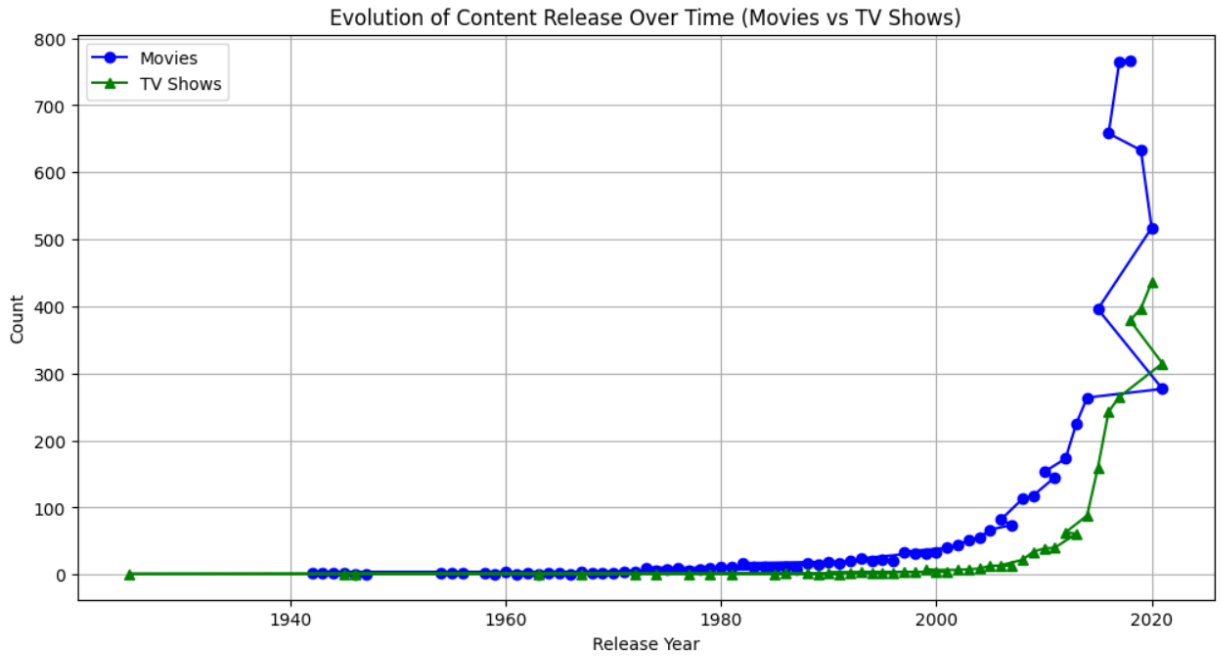
To visualize the evolution of content release over time by comparing the number of Movies vs TV Shows released each year using line plots, helping identify trends and differences in content production between Movies and TV Shows.

Step:-

1. Filter Movies and TV Shows
2. Count Movies and TV Shows Released Each Year
3. Plot the Data

Code:-

```
df_movies = df[df['type']=='Movie']
df_tv_shows = df[df['type']=='TV Show']

movies_per_year = df_movies["release_year"].value_counts()
tv_shows_per_year = df_tv_shows["release_year"].value_counts()

# Plot the data
plt.figure(figsize=(12, 6))

# Plot for Movies and TV Shows
plt.plot(movies_per_year.index, movies_per_year.values, marker='o', label='Movies', color='blue')
plt.plot(tv_shows_per_year.index, tv_shows_per_year.values, marker='^', label='TV Shows', color='green')

# Add Titles and Labels
plt.title('Evolution of Content Release Over Time (Movies vs TV Shows)')
plt.xlabel('Release Year')
plt.ylabel('Count')
plt.legend()
plt.grid(True)

plt.show()
```



Explanation:-

- Helps visualize the evolution of Movies and TV Shows over time. Allows for an easy comparison of the growth trends between Movies and TV Shows.
- Useful for identifying shifts in content production and understanding media consumption trends

Trend of Content Added to Netflix by Year-

Objective:-

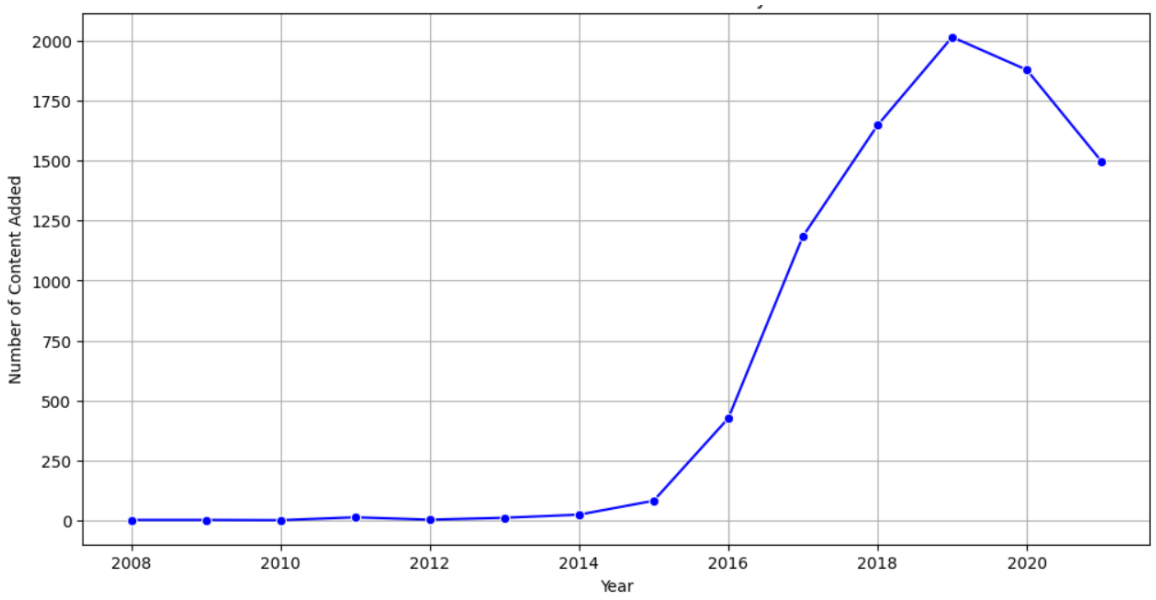
To visualize the trend of content additions to Netflix over time by year, helping to track how the volume of content added has changed across the years.

Step:-

1. Extract Year and Month from “Date_added”
2. Group by Year and Count the Number of Additions
3. Plot the Trend of Content Additions by Year

Code:-

```
# Extract year and month from 'Date_added'
df['Year_added'] = df['date_added'].dt.year
df['Month_added'] = df['date_added'].dt.month

# Group by year and count the number of additions
content_per_year = df['Year_added'].value_counts().sort_index()

# Plot the trend of content additions by year
plt.figure(figsize=(12, 6))
sns.lineplot(x=content_per_year.index, y=content_per_year.values, marker='o', color='blue')
plt.title('Trend of Content Added to Netflix by Year')
plt.xlabel('Year')
plt.ylabel('Number of Content Added')
plt.grid(True)
plt.show()
```



Explanation:-

- Helps understand how the volume of content added to Netflix has changed over the years.
- Provides insights into content addition trends which can be useful for business, marketing, and content strategy decisions.
- Shows how Netflix's content library has grown over time

Heatmap of Netflix Content Added by Year and Month-

Objective:-

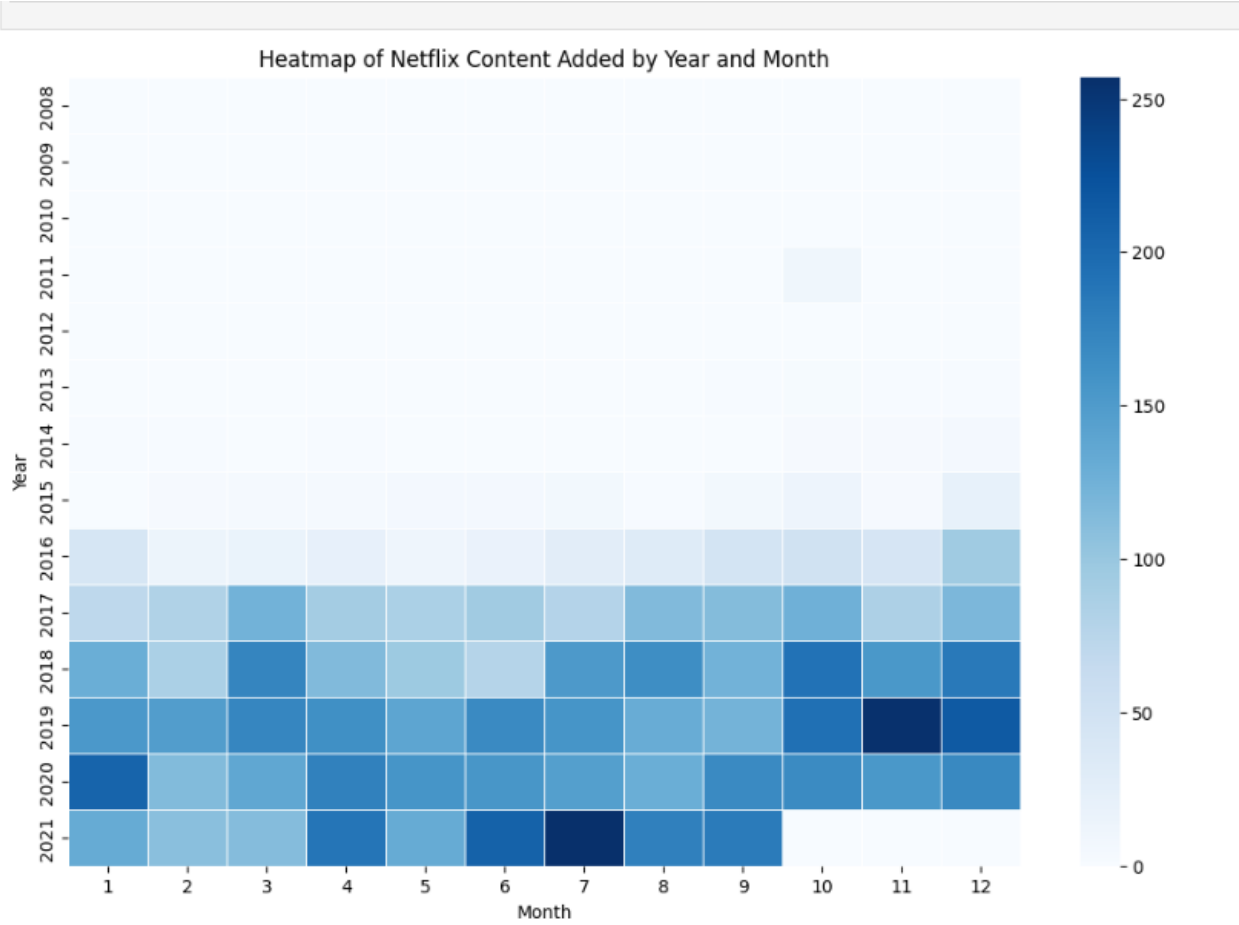
To visualize monthly trends in content additions to Netflix using a heatmap, showing how content additions vary across months and years.

Step:-

1. Group Data by Year and Month
2. Create a Heatmap for Monthly Trends

Code:-

```
]: # Analyze trends by year and month
content_per_month_year = df.groupby(['Year_added', 'Month_added']).size().unstack(fill_value=0)

# Heatmap for monthly trends
plt.figure(figsize=(12, 8))
sns.heatmap(content_per_month_year, cmap='Blues', linewidths=0.5)
plt.title('Heatmap of Netflix Content Added by Year and Month')
plt.xlabel('Month')
plt.ylabel('Year')
plt.show()
```



Explanation:-

- Helps to identify seasonal trends in content addition over the years (e.g., if more content is added during certain months)
- Provides insights into Netflix's content release strategy and potential periods of high activity.
- Enables quick spotting of patterns, such as monthly surges in content addition.

Distribution of Lag Between Release Year and Date Added-

Objective:-

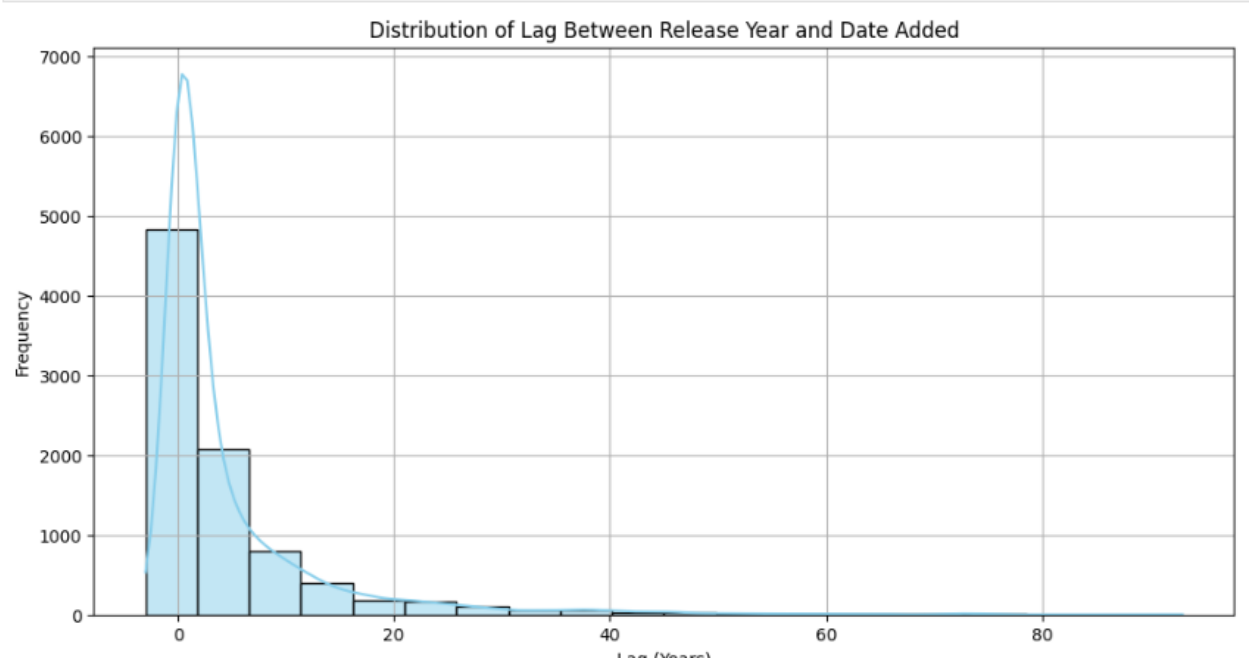
To visualize the distribution of the lag between the release year and the date added to Netflix, showing how many years typically pass before content is added to the platform

Step:-

1. Extract Year from 'Date_added' and Calculate Lag
2. Plot the Distribution of the Lag
3. Add Titles, Labels, and Gridlines

Code:-

```
# Extract year from 'Date_added' and calculate lag
df['Year_added'] = df['date_added'].dt.year
df['Lag'] = df['Year_added'] - df['release_year']

# Plot the distribution of the lag
plt.figure(figsize=(12, 6))
sns.histplot(df['Lag'], bins=20, kde=True, color='skyblue')
plt.title('Distribution of Lag Between Release Year and Date Added')
plt.xlabel('Lag (Years)')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```



Explanation:-

- Helps to understand the typical lag between content release and when it is added to Netflix.
- Provides insights into Netflix's content acquisition strategy (e.g., licensing or waiting period before content is added).
- Shows if there's a consistent delay or faster addition process for different types of content.

Lag Between Release Year and Date Added by Type-

Objective:-

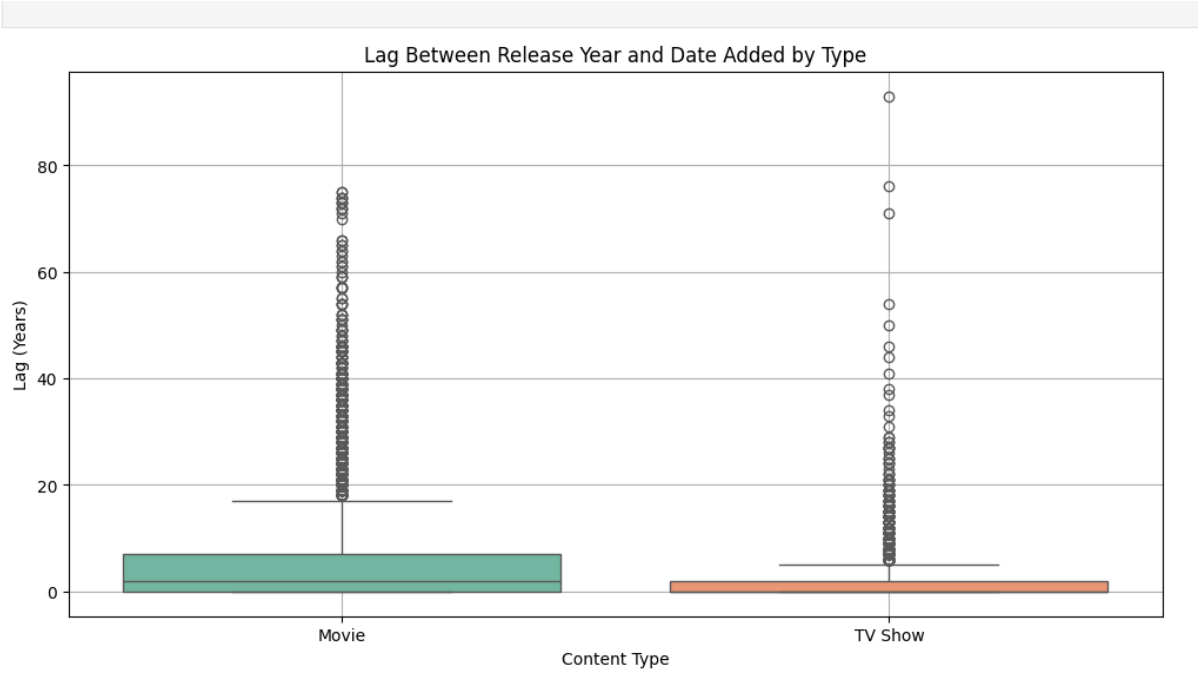
To visualize the distribution of lag (time between release year and date added) for Movies vs TV Shows using a boxplot, allowing for a comparison of how the lag differs across content types.

Step:-

1. Create a Boxplot for Lag by Type (Movies vs TV Shows)
2. Add Titles, Labels, and Gridlines

Code:-

```
]:  # Boxplot for lag by type (Movies vs TV Shows)
plt.figure(figsize=(12, 6))
sns.boxplot(x='type', y='Lag', data=df, palette='Set2')
plt.title('Lag Between Release Year and Date Added by Type')
plt.xlabel('Content Type')
plt.ylabel('Lag (Years)')
plt.grid(True)
plt.show()
```



Explanation:-

1. Allows for a comparison of lag between Movies and TV Shows on Netflix.
2. Helps identify patterns in how quickly content is added after its release, and if there's a significant difference between the two content types.
3. Boxplots show the median, spread, and outliers, offering a deeper understanding of the lag distribution.

Content Categorization:-

Top 10 Popular Genres for Movies on Netflix-

Objective:-

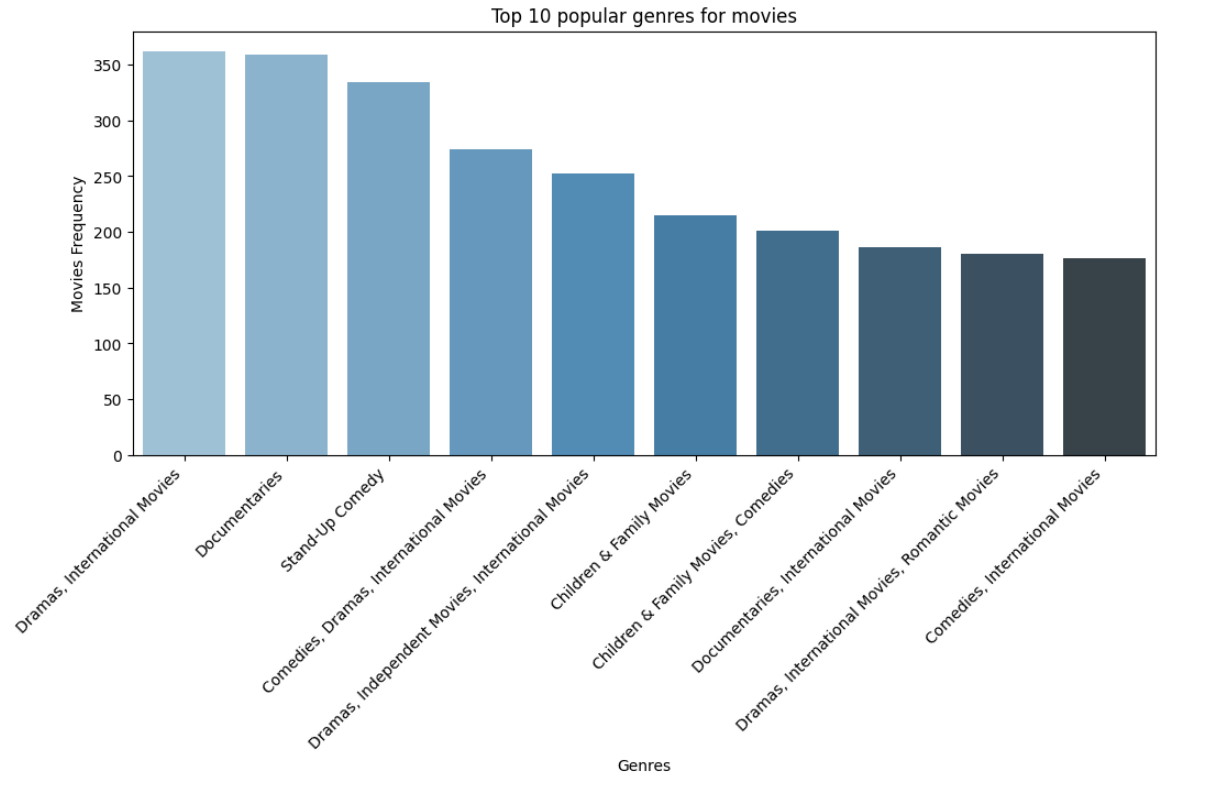
To visualize the top 10 most popular movie genres based on their frequency of occurrence on Netflix, helping to understand the most common genres for movies.

Step:-

1. Get the Top 10 Most Popular Movie Genres
2. Create the Barplot for Top 10 Popular Movie Genres
3. Customize the X-axis and Add Label

Code:-

```
# Get the top 10 most popular movie genres
popular_movie_genre=df[df['type']=='Movie'].groupby("listed_in").size().sort_values(ascending=False)[:10]

plt.figure(figsize=(12, 5))
sns.barplot(x=popular_movie_genre.index, y=popular_movie_genre.values, palette='Blues_d')
plt.xticks(rotation=45, ha='right')

# Add labels and title
plt.xlabel("Genres")
plt.ylabel("Movies Frequency")
plt.title("Top 10 popular genres for movies")
plt.show()
```



Explanation:-

- Helps to identify the **most popular movie genres** on Netflix.
- Provides insights into Netflix's **genre distribution**, which can be useful for recommendations, content creation, and understanding viewer preferences.
- The **top 10 genres** visualization helps pinpoint **dominant trends** in movie content.

Top 10 Popular Genres for TV Shows on Netflix-

Objective:-

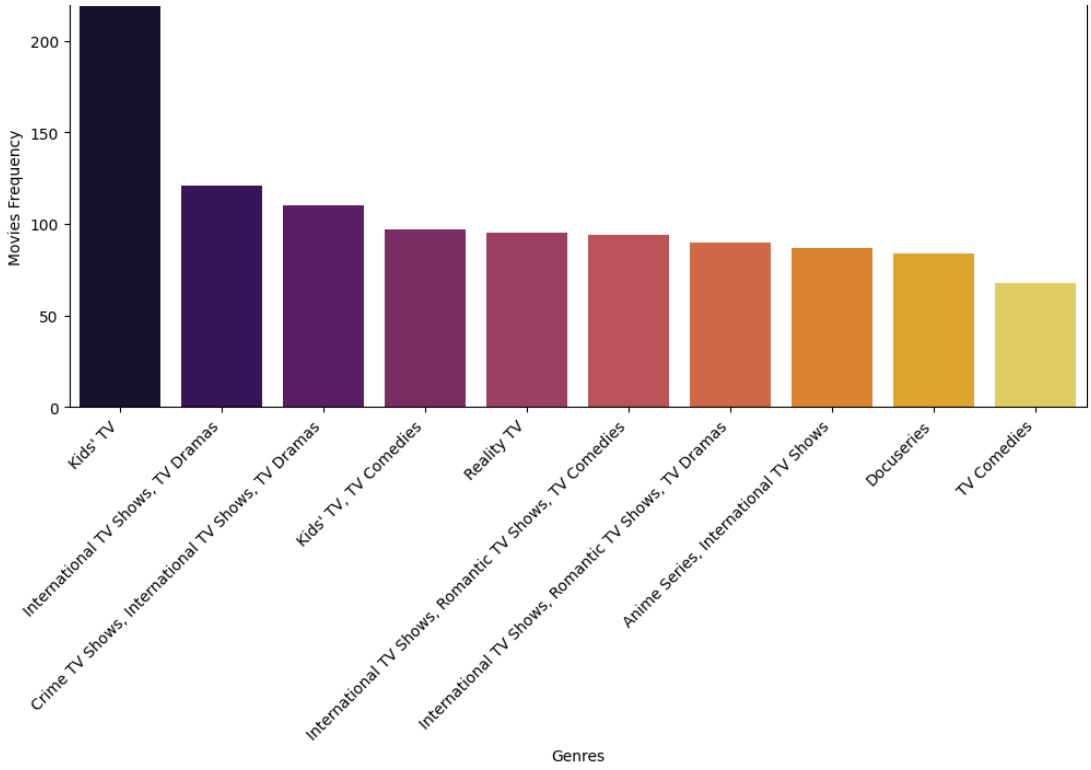
To visualize the top 10 most popular genres for TV shows (series) on Netflix, providing insights into the most frequent genres for series content.

Step:-

1. Get the Top 10 Most Popular TV Show Genres
2. Create the Barplot for Top 10 Popular TV Show Genres

Code:-

```
5]: # Get the top 10 most popular series genres

popular_series_genre=df[df['type']=='TV Show'].groupby("listed_in").size().sort_values(ascending=False)[:10]
plt.figure(figsize=(12, 5))
sns.barplot(x=popular_series_genre.index, y=popular_series_genre.values, palette='inferno')
plt.xticks(rotation=45, ha='right')

# Add labels and title
plt.xlabel("Genres")
plt.ylabel("Movies Frequency")
plt.title("Top 10 popular genres for Series")
plt.show()
```



Explanation:-

- Helps identify the most popular genres for TV shows on Netflix.
- Provides insights into the genre distribution for series content, which is valuable for content analysis, recommendations, and viewer preferences.
- The top 10 genres visualization offers a quick overview of dominant TV show genres on the platform.

Rating Distribution:-

Distribution of Ratings on Netflix-

Objective:-

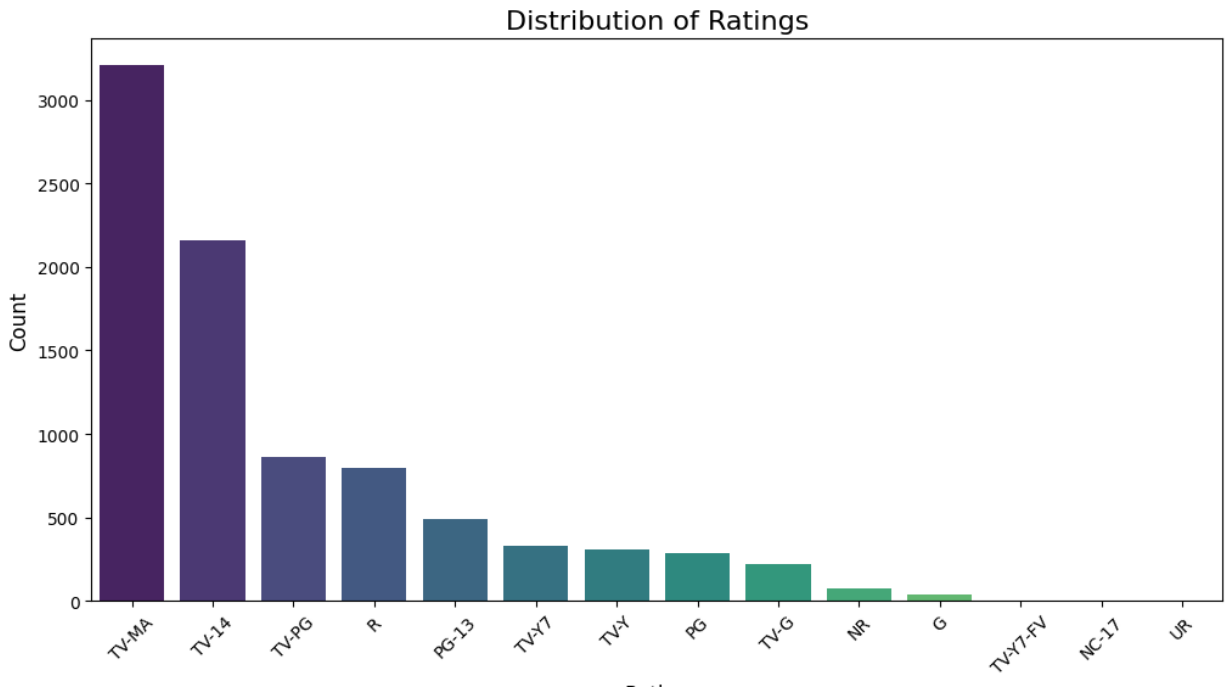
To visualize the distribution of ratings for the content on Netflix, providing insights into how different ratings are distributed across movies and TV shows.

Step:-

1. Plot the Bar Chart for Ratings Distribution
2. Add Title, Labels, and Rotate X-axis Labels

Code:-

```
]: # Plot the bar chart
plt.figure(figsize=(12, 6))
sns.barplot(x=rating.index, y=rating.values, palette='viridis')

plt.title('Distribution of Ratings', fontsize=16)
plt.xlabel('Rating', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.xticks(rotation=45)
plt.show()
```



Explanation:-

- Provides insights into how Netflix's content is distributed across different ratings, helping understand viewer preferences.
- Visualizes the frequency of each rating category, highlighting if certain ratings are more common than others.

- Can help identify if most content is rated highly or has a wide range of ratings.

Rating Distribution Comparison: Movies vs TV Shows on Netflix-

Objective:-

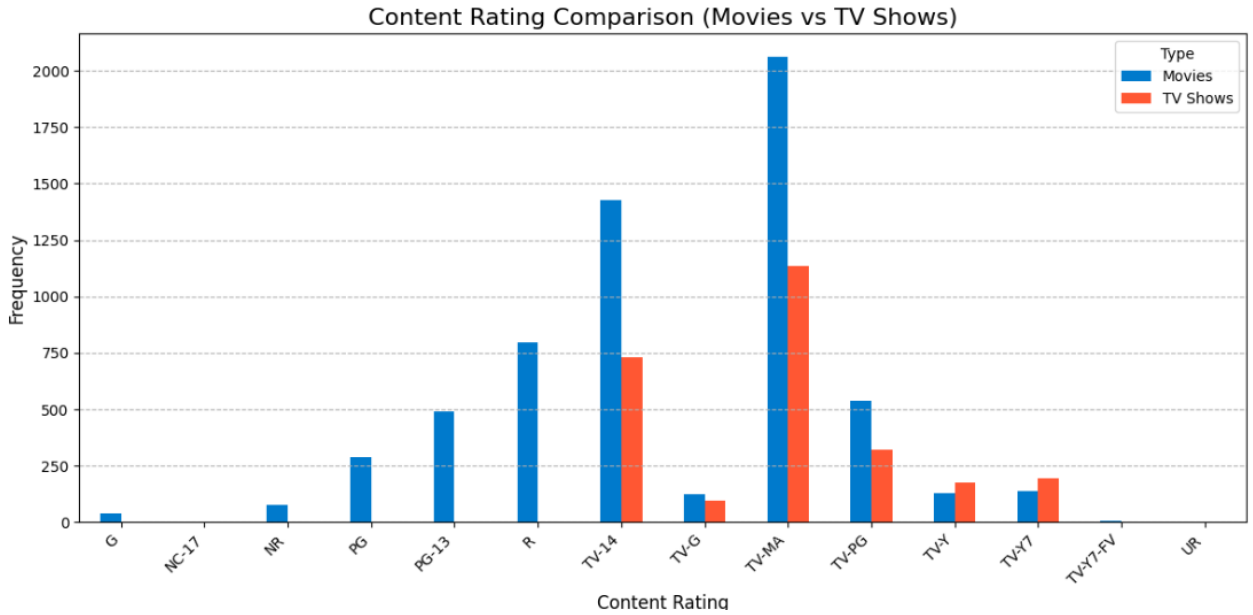
To compare the distribution of ratings for Movies vs TV Shows on Netflix, providing insights into how the ratings differ between these two types of content.

Step:-

1. Separate Movies and TV Shows
2. Count Ratings for Movies and TV Shows
3. Combine Ratings Data for Comparison
4. Plot the Side-by-Side Bar Chart

Code:-

```
# Separate Movies and TV Shows
movies = df[df['type'] == 'Movie']
tv_shows = df[df['type'] == 'TV Show']

# Count ratings for Movies
movie_ratings = movies['rating'].value_counts()

# Count ratings for TV Shows
tv_show_ratings = tv_shows['rating'].value_counts()

# Combine into one DataFrame for comparison
ratings_comparison = pd.DataFrame({'Movies': movie_ratings,'TV Shows': tv_show_ratings})
# Plot side-by-side bar chart
ratings_comparison.plot(kind='bar', figsize=(12, 6), color=['#007ACC', '#FF5733'])
plt.title('Content Rating Comparison (Movies vs TV Shows)', fontsize=16)
plt.xlabel('Content Rating', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.legend(title='Type')
plt.grid(axis='y', linestyle='--')
plt.tight_layout()
plt.show()
```



Explanation:-

- Allows for a direct comparison between the ratings of Movies and TV Shows on Netflix.
- Helps to identify if there are differences in how Movies and TV Shows are rated by viewers
- Provides insights into which content type may have more favorable ratings or if one tends to have higher ratings than the other.

Duration Aspect:-

Average Duration of Movies vs TV Shows on Netflix-

Objective:-

To compare the average duration of Movies and TV Shows on Netflix by calculating and visualizing their typical duration in minutes (for Movies) and seasons (for TV Shows).

Step:-

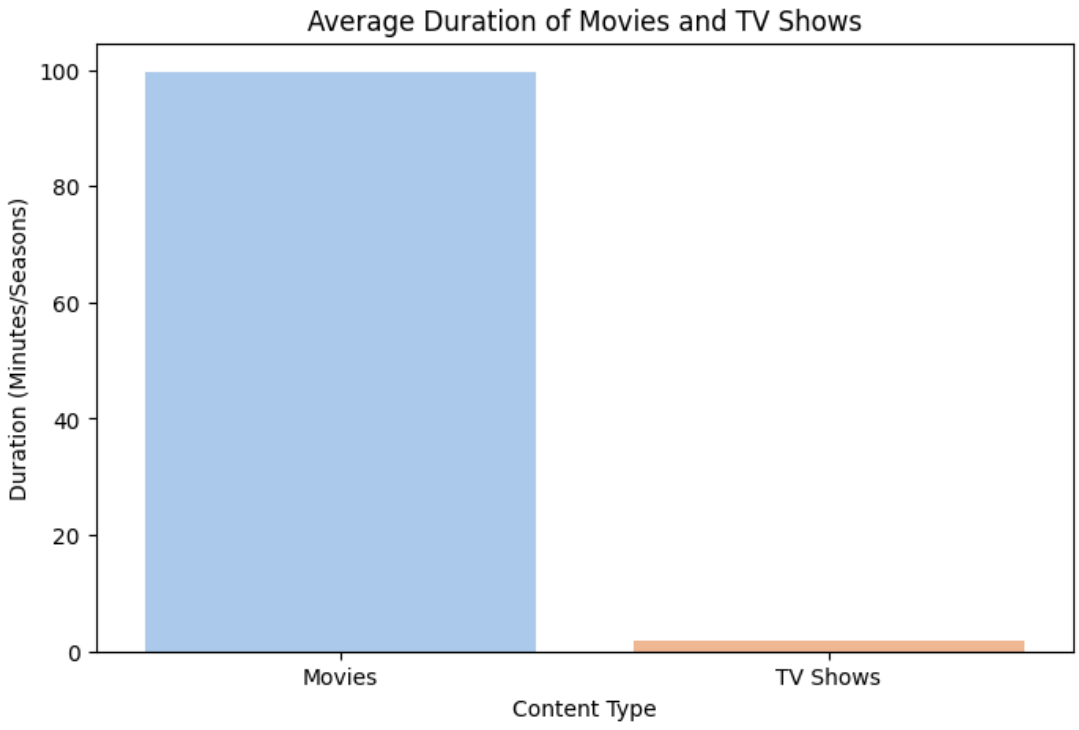
1. Handle Missing Values
2. Separate Movies and TV Shows
3. Extract Duration for Movies
4. Extract Duration for TV Shows

5. Calculate Average Duration
6. Visualization

Code:-

```
df['duration'] = df['duration'].fillna('0')  # Fill missing values for safety

# Separate data for Movies and TV Shows
movies = df[df['type'] == 'Movie']
tv_shows = df[df['type'] == 'TV Show']

# Extract numeric duration for Movies
movies['duration_minutes'] = movies['duration'].str.extract('(\d+)').astype(float)

# Extract numeric duration for TV Shows
tv_shows['duration_seasons'] = tv_shows['duration'].str.extract('(\d+)').astype(float)

# Calculate average duration
average_movie_duration = movies['duration_minutes'].mean()
average_tv_show_seasons = tv_shows['duration_seasons'].mean()

# Print results
print(f"Average Movie Duration: {average_movie_duration:.2f} minutes")
print(f"Average TV Show Duration: {average_tv_show_seasons:.2f} seasons")

# Visualization
plt.figure(figsize=(8, 5))
sns.barplot(x=['Movies', 'TV Shows'], y=[average_movie_duration, average_tv_show_seasons],palette='pastel')
plt.title('Average Duration of Movies and TV Shows')
plt.ylabel('Duration (Minutes/Seasons)')
plt.xlabel('Content Type')
plt.show()
```

```
Average Movie Duration: 99.59 minutes
Average TV Show Duration: 1.75 seasons
```



Explanation:-

- Provides insights into the typical duration of content on Netflix, helping users understand the difference in how Movies and TV Shows are structured.
- Can be useful for users to decide what type of content they prefer based on time commitment
- Provides a better understanding of Netflix's content library, highlighting whether movies or TV shows are generally longer in terms of duration

Analysis of Short Films, Long Movies, and TV Shows on Netflix-

Objective:-

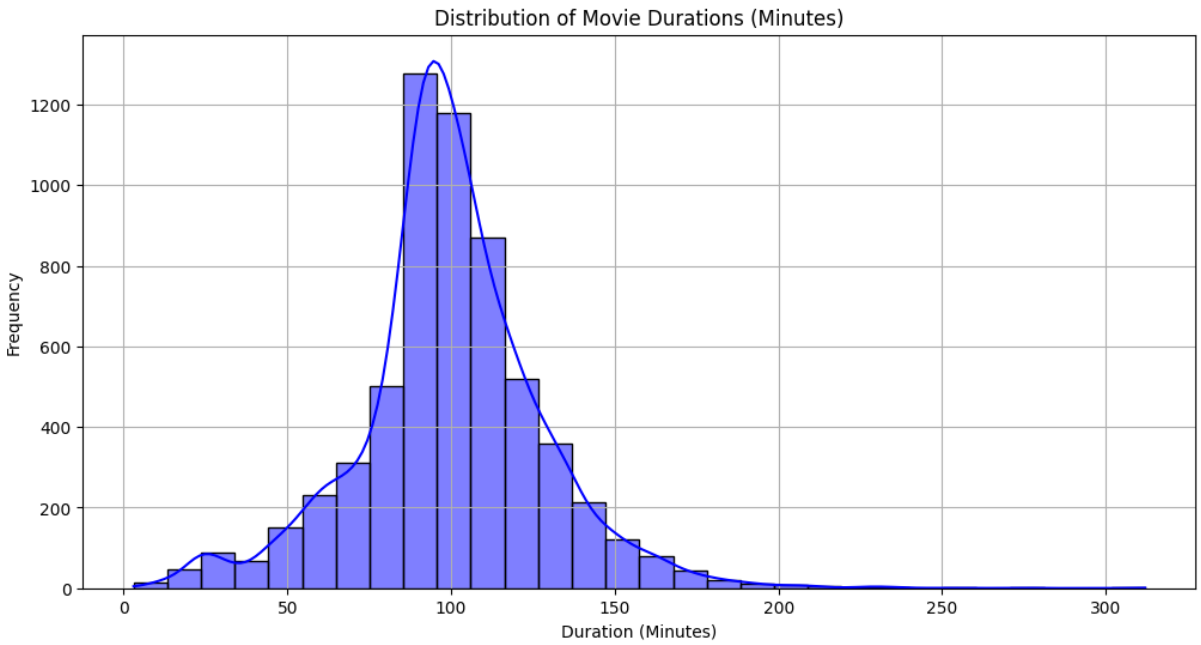
The objective of this analysis is to examine the duration of content on Netflix**,** including identifying short films, long movies, and long-running TV shows**.** This helps in understanding content distribution patterns and user preferences based on movie length and TV show seasons.

Step:-

1. Identifying Short and Long Content
2. Displaying Key Insights
3. Visualizing the Movie Duration Distribution

Code:-

```
# Identify short and long content
short_films = movies[movies['duration_minutes'] < 40]
long_movies = movies[movies['duration_minutes'] > 180]
long_tv_shows = tv_shows[tv_shows['duration_seasons'] > 10]

# Print insights
print(f"Number of Short Films (<40 min): {len(short_films)}")
print(f"Number of Long Movies (>180 min): {len(long_movies)}")


# Plot distributions for movies
plt.figure(figsize=(12, 6))
sns.histplot(movies['duration_minutes'], bins=30, kde=True, color='blue')
plt.title('Distribution of Movie Durations (Minutes)')
plt.xlabel('Duration (Minutes)')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```



Explanation:-

- Helps understand the diversity of content on Netflix.
- Useful for users who prefer short films vs. long movies.
- Valuable for content creators and analysts to see trends in content duration.

Analysis of Long-Running TV Shows and Their Season Distribution on Netflix

Objective:-

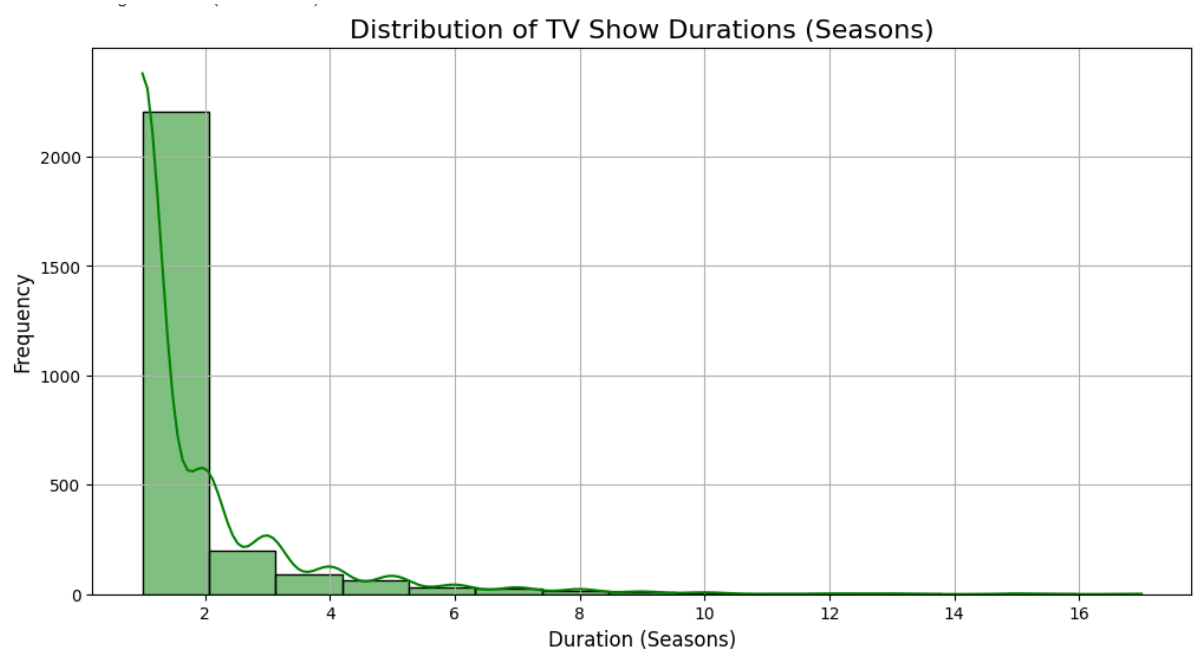
To analyze the distribution of TV show durations (in seasons) on Netflix and identify long-running series with more than 10 seasons.

Step:-

1. Identifying Long-Running TV Shows
2. Visualizing the Season Distribution of TV Shows

Code:-

```
print(f"Number of Long TV Shows (>10 seasons): {len(long_tv_shows)}")
# Plot distributions for TV shows
plt.figure(figsize=(12, 6))
sns.histplot(tv_shows['duration_seasons'], bins=15, kde=True, color='green')
plt.title('Distribution of TV Show Durations (Seasons)', fontsize=16)
plt.xlabel('Duration (Seasons)', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.grid(True)
plt.show()
```



Explanation:-

- Helps in understanding the lifespan of TV shows on Netflix.
- Useful for content creators to see if Netflix prefers shorter vs. long-running series.
- Provides insights into user preferences for binge-worthy TV shows.

Conclusion of Netflix Content Analysis Project:-

Summary of Findings-

This project involved analyzing Netflix's dataset to uncover insights into content distribution, trends, and patterns across movies and TV shows. The analysis covered multiple aspects, including content type, release trends, genre popularity, content duration, and ratings distribution.

Key Insights-

Content Trends & Time Series Analysis

- Netflix's content additions have increased significantly in recent years, with a peak in content additions around 2018–2020.
- A time lag exists between the original release year and when content is added to Netflix, with most content being added 1–5 years after its release.

Genre Popularity & Category Distribution

- Drama, Comedy, and Action are among the most popular genres for movies.
- For TV Shows, International TV Dramas and Crime Thrillers are highly preferred.
- Netflix's content caters to a global audience, with top content-producing countries USA, India, and the UK including the USA, India, and the UK.

Ratings & Audience Classification

- The majority of movies and TV shows fall under ratings like TV-MA, TV-14, and PG-13, indicating that Netflix primarily targets mature and young-adult audiences.
- A comparison of movie vs. TV show ratings shows that movies have a more diverse range of ratings, while TV shows tend to cluster around mature ratings (TV-MA, TV-14).

Content Duration & Viewing Preferences

- Most movies are between 80–120 minutes, with very few exceeding 180 minutes.
- TV shows are mostly short, with the majority having 1–3 seasons, while only a few exceed 10 seasons.
- The presence of short films (<40 min) suggests Netflix also includes documentaries and animated shorts in its catalog

Final Thoughts

This analysis provides valuable insights into Netflix's content trends, audience preferences**,** and streaming strategies. By leveraging these insights, Netflix can make data-driven decisions regarding content acquisition, production, and marketing to enhance viewer satisfaction and global reach.