netflixdataanalysis

September 20, 2024

[3]:
```python
# Step 1: Import Required Libraries
import pandas as pd
```

[5]:
```python
# Step 2: Load the Dataset
file_path = r"C:\Users\Admin\Downloads\netflix1.csv"
netflix_df = pd.read_csv(file_path)

# Display the first few rows of the dataset to understand its structure
netflix_df.head()
```

[5]:
```
  show_id     type                             title         director  \
0      s1    Movie              Dick Johnson Is Dead  Kirsten Johnson
1      s3  TV Show                         Ganglands  Julien Leclercq
2      s6  TV Show                     Midnight Mass    Mike Flanagan
3     s14    Movie  Confessions of an Invisible Girl    Bruno Garotti
4      s8    Movie                           Sankofa     Haile Gerima

         country date_added  release_year rating  duration  \
0  United States  9/25/2021          2020  PG-13    90 min
1         France  9/24/2021          2021  TV-MA  1 Season
2  United States  9/24/2021          2021  TV-MA  1 Season
3         Brazil  9/22/2021          2021  TV-PG    91 min
4  United States  9/24/2021          1993  TV-MA   125 min

                                            listed_in
0                                       Documentaries
1  Crime TV Shows, International TV Shows, TV Act…
2                  TV Dramas, TV Horror, TV Mysteries
3                  Children & Family Movies, Comedies
4   Dramas, Independent Movies, International Movies
```

1 Data Cleaning

[10]:
```python
# Step 3: Data Cleaning

# Check for missing values
missing_values = netflix_df.isnull().sum()
```

```
missing_values
```

[10]:
```
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
```

[12]:
```
# Remove duplicates (if any)
netflix_df.drop_duplicates(inplace=True)
```

[14]:
```
# Convert 'date_added' to datetime format
netflix_df['date_added'] = pd.to_datetime(netflix_df['date_added'])
```

[16]:
```
# Clean up 'duration' column: separate minutes from seasons for movies and TV␣
 ↪shows
def clean_duration(row):
    if 'Season' in row:
        return 'TV Show'
    else:
        return row.replace(' min', '')
```

[18]:
```
# Apply the cleaning function to the duration column
netflix_df['duration_cleaned'] = netflix_df['duration'].apply(clean_duration)

# Convert duration to numeric where appropriate
netflix_df['duration_cleaned'] = pd.to_numeric(netflix_df['duration_cleaned'],␣
 ↪errors='coerce')

# Drop columns that are unnecessary for analysis ('show_id' can be dropped as␣
 ↪it's just an ID)
netflix_df_cleaned = netflix_df.drop(columns=['show_id'])
```

[20]:
```
# Display summary of missing values after cleaning and the first few rows of␣
 ↪cleaned data
cleaned_missing_values = netflix_df_cleaned.isnull().sum()
cleaned_head = netflix_df_cleaned.head()

cleaned_missing_values, cleaned_head
```

```
[20]: (type                  0
 title                 0
 director              0
 country               0
 date_added            0
 release_year          0
 rating                0
 duration              0
 listed_in             0
 duration_cleaned    2664
 dtype: int64,
       type                              title          director        country  \
 0    Movie               Dick Johnson Is Dead  Kirsten Johnson  United States
 1  TV Show                          Ganglands  Julien Leclercq         France
 2  TV Show                      Midnight Mass    Mike Flanagan  United States
 3    Movie  Confessions of an Invisible Girl    Bruno Garotti         Brazil
 4    Movie                            Sankofa     Haile Gerima  United States

   date_added  release_year rating  duration  \
 0 2021-09-25          2020  PG-13    90 min
 1 2021-09-24          2021  TV-MA  1 Season
 2 2021-09-24          2021  TV-MA  1 Season
 3 2021-09-22          2021  TV-PG    91 min
 4 2021-09-24          1993  TV-MA   125 min

                                          listed_in  duration_cleaned
 0                                    Documentaries              90.0
 1  Crime TV Shows, International TV Shows, TV Act…               NaN
 2                TV Dramas, TV Horror, TV Mysteries               NaN
 3                Children & Family Movies, Comedies              91.0
 4   Dramas, Independent Movies, International Movies            125.0  )
```

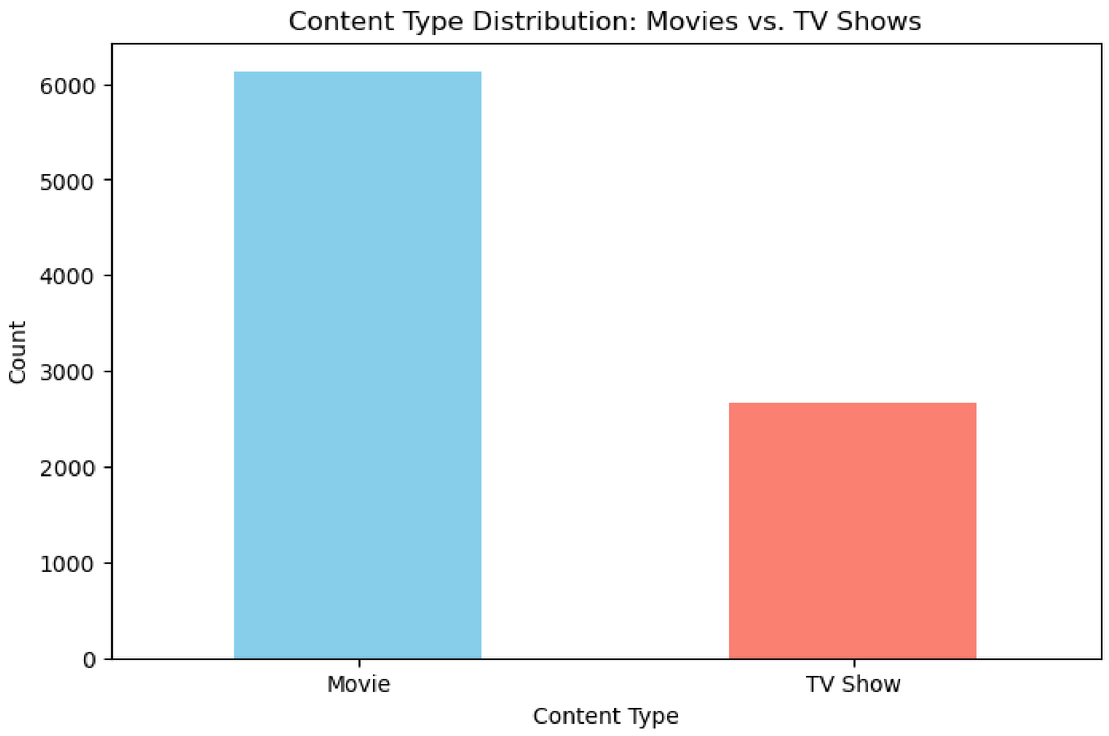
2 Exploratory Data Analysis

```
[23]: # Step 4: EDA - Content Type Distribution (Movies vs. TV Shows)

# Calculate the distribution of content types
content_type_distribution = netflix_df_cleaned['type'].value_counts()

# Plot the distribution
import matplotlib.pyplot as plt

plt.figure(figsize=(8,5))
content_type_distribution.plot(kind='bar', color=['skyblue', 'salmon'])
plt.title('Content Type Distribution: Movies vs. TV Shows')
plt.ylabel('Count')
plt.xlabel('Content Type')
plt.xticks(rotation=0)
plt.show()
```



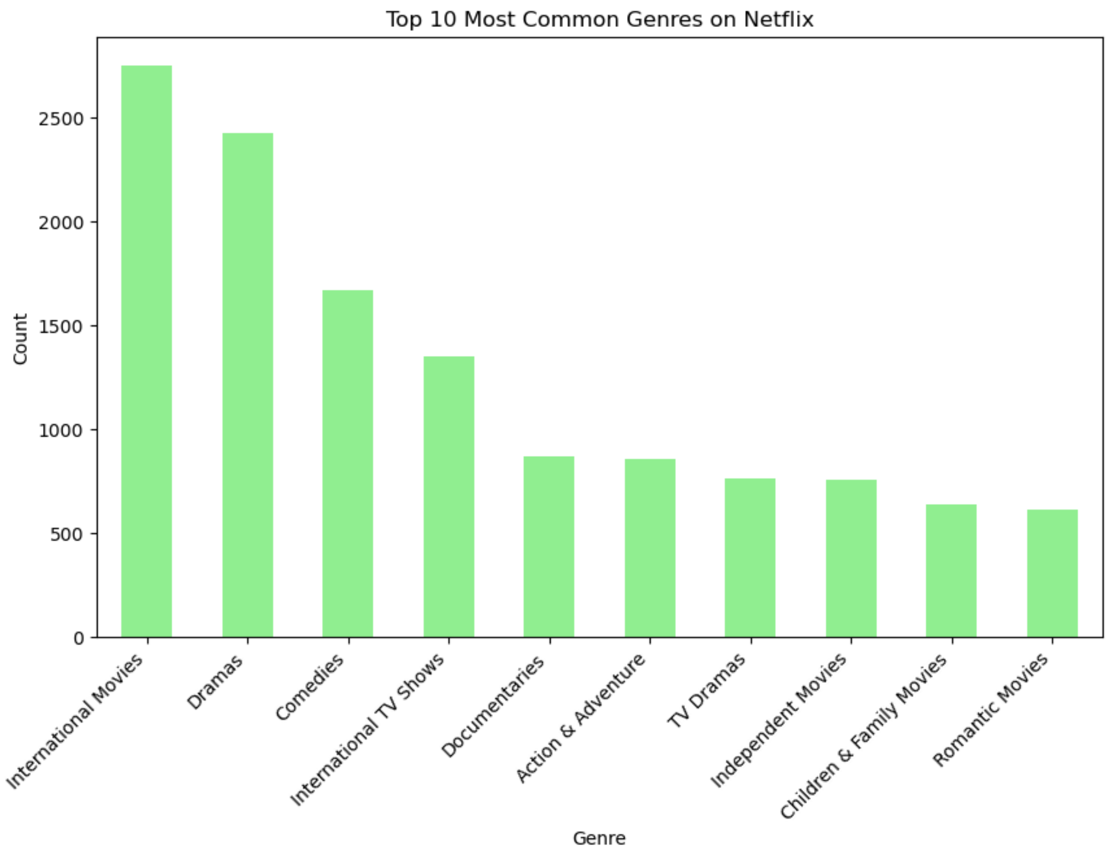
[29]:
```
# Step 4: EDA - Most Common Genres

# Split the 'listed_in' column to extract individual genres
genres = netflix_df_cleaned['listed_in'].str.split(', ', expand=True).stack()

# Calculate the frequency of each genre
most_common_genres = genres.value_counts().head(10)

# Plot the most common genres
plt.figure(figsize=(10,6))
most_common_genres.plot(kind='bar', color='lightgreen')
plt.title('Top 10 Most Common Genres on Netflix')
plt.ylabel('Count')
plt.xlabel('Genre')
plt.xticks(rotation=45, ha='right')
plt.show()

most_common_genres
```



[29]:
```
International Movies        2752
Dramas                      2426
Comedies                    1674
International TV Shows      1349
Documentaries                869
Action & Adventure           859
TV Dramas                    762
Independent Movies           756
Children & Family Movies     641
Romantic Movies              616
Name: count, dtype: int64
```

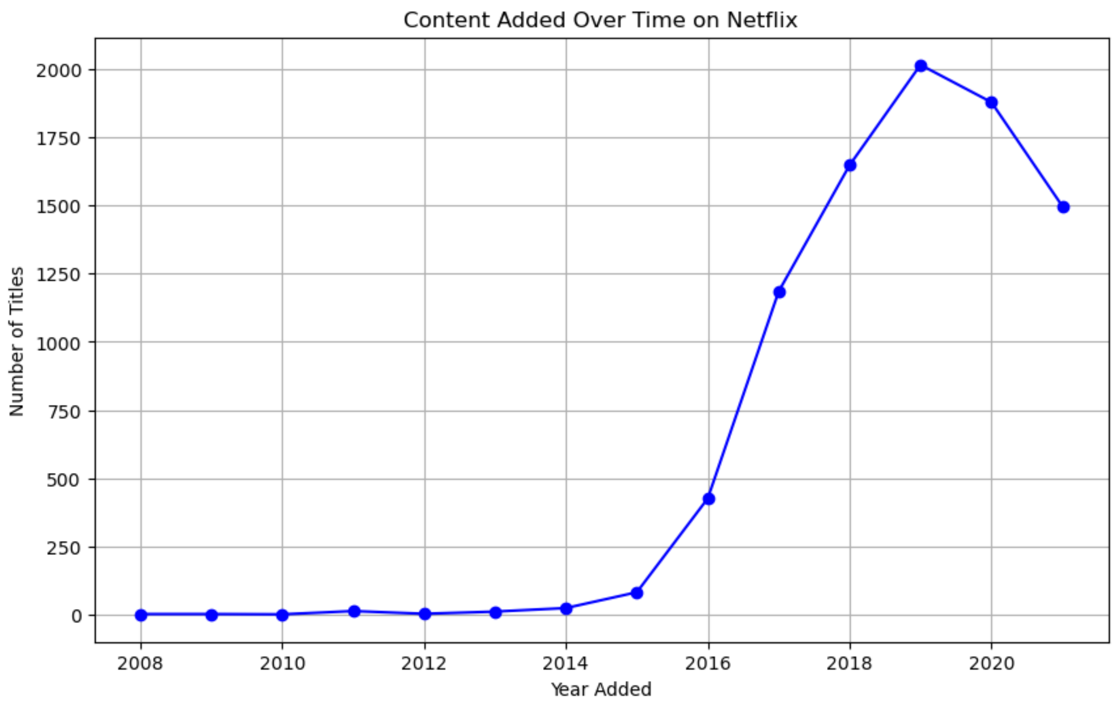
[34]:
```python
# Step 4: EDA - Content Added Over Time

# Extract the year from the 'date_added' column for analysis
netflix_df_cleaned['year_added'] = netflix_df_cleaned['date_added'].dt.year

# Count the number of titles added per year
content_added_over_time = netflix_df_cleaned['year_added'].value_counts().
↪sort_index()
```

```
# Plot the trend of content added over time
plt.figure(figsize=(10,6))
content_added_over_time.plot(kind='line', marker='o', color='blue')
plt.title('Content Added Over Time on Netflix')
plt.ylabel('Number of Titles')
plt.xlabel('Year Added')
plt.grid(True)
plt.show()

content_added_over_time
```



[34]:
```
year_added
2008       2
2009       2
2010       1
2011      13
2012       3
2013      11
2014      24
2015      82
2016     426
2017    1185
2018    1648
```

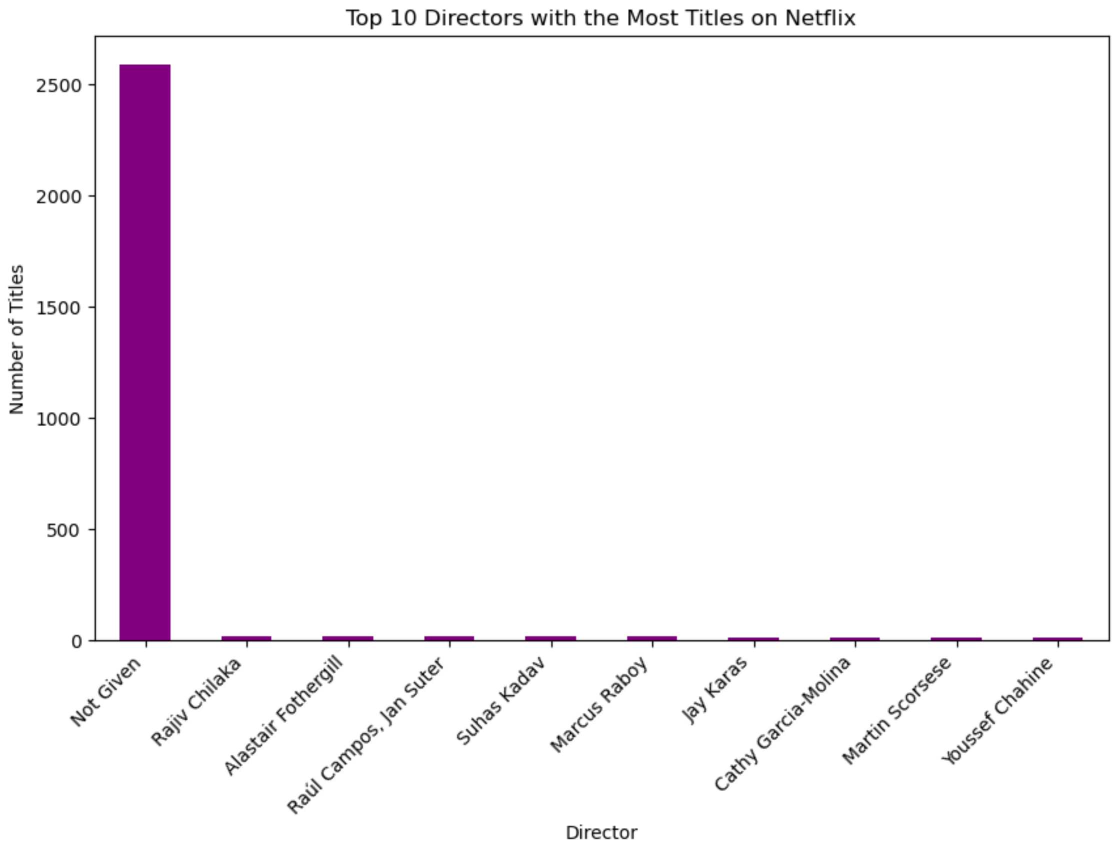
```
2019    2016
2020    1879
2021    1498
Name: count, dtype: int64
```

[38]:
```
# Step 4: EDA - Top 10 Directors with the Most Titles

# Count the number of titles for each director
top_10_directors = netflix_df_cleaned['director'].value_counts().head(10)

# Plot the top 10 directors
plt.figure(figsize=(10,6))
top_10_directors.plot(kind='bar', color='purple')
plt.title('Top 10 Directors with the Most Titles on Netflix')
plt.ylabel('Number of Titles')
plt.xlabel('Director')
plt.xticks(rotation=45, ha='right')
plt.show()

top_10_directors
```



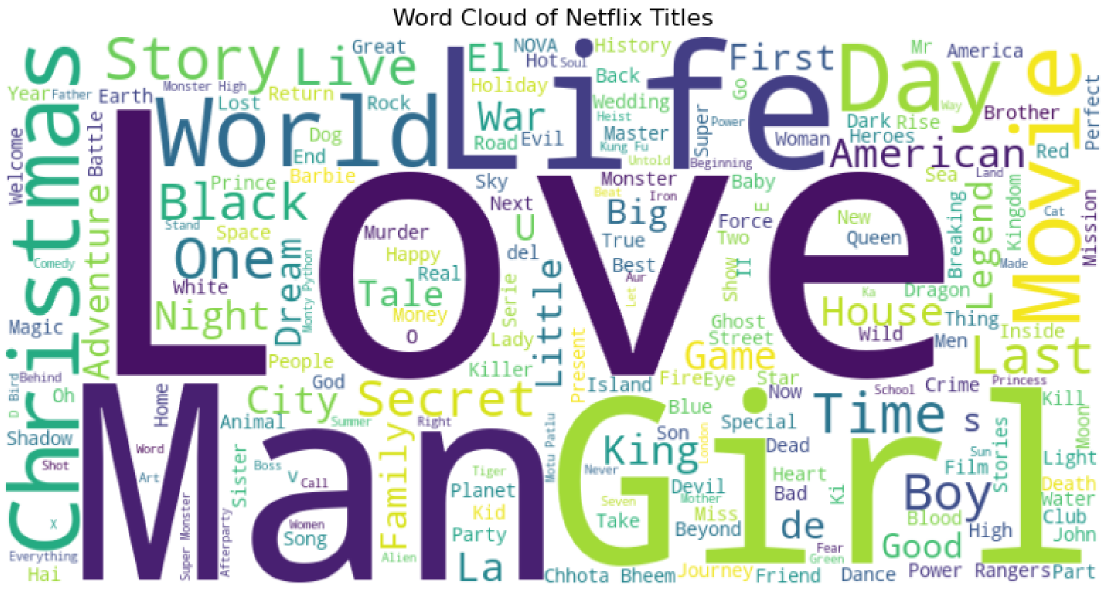
```
[38]: director
      Not Given                2588
      Rajiv Chilaka              20
      Alastair Fothergill        18
      Raúl Campos, Jan Suter     18
      Suhas Kadav                16
      Marcus Raboy               16
      Jay Karas                  14
      Cathy Garcia-Molina        13
      Martin Scorsese            12
      Youssef Chahine            12
      Name: count, dtype: int64
```

```python
[49]: #!pip install wordcloud

      # Step 4: EDA - Word Cloud of Movie Titles
      from wordcloud import WordCloud

      # Generate a word cloud for the titles
      title_text = ' '.join(netflix_df_cleaned['title'].dropna().values)
      wordcloud = WordCloud(width=800, height=400, background_color='white').
        ↪generate(title_text)

      # Plot the word cloud
      plt.figure(figsize=(10,6))
      plt.imshow(wordcloud, interpolation='bilinear')
      plt.axis('off')
      plt.title('Word Cloud of Netflix Titles')
      plt.show()
```



3 Feature Engineering

[51]:
```
# Feature Engineering: Creating new features

# 1. Number of Genres: Count how many genres each title has
netflix_df_cleaned['num_genres'] = netflix_df_cleaned['listed_in'].apply(lambda␣
↪x: len(x.split(', ')))

# 2. Duration in Minutes: Keep the cleaned duration for movies and handle TV␣
↪shows as NaN or "Seasons"
netflix_df_cleaned['duration_in_minutes'] = netflix_df_cleaned.apply(
    lambda row: row['duration_cleaned'] if row['type'] == 'Movie' else None,␣
↪axis=1)

# 3. Year Difference: Calculate the difference between release year and year␣
↪added
netflix_df_cleaned['year_diff'] = netflix_df_cleaned['year_added'] -␣
↪netflix_df_cleaned['release_year']

# Display the first few rows to check the newly engineered features
netflix_df_cleaned[['title', 'num_genres', 'duration_in_minutes', 'year_diff']].
↪head()
```

[51]:
```
                              title  num_genres  duration_in_minutes  \
0              Dick Johnson Is Dead           1                 90.0
1                         Ganglands           3                  NaN
2                     Midnight Mass           3                  NaN
3  Confessions of an Invisible Girl           2                 91.0
4                           Sankofa           3                125.0

   year_diff
0          1
1          0
2          0
3          0
4         28
```

[53]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity

# Step 1: Combine relevant features (genres, director, title) into a single␣
↪string for each content
netflix_df_cleaned['combined_features'] = netflix_df_cleaned.apply(
    lambda row: f"{row['title']} {row['listed_in']} {row['director']}", axis=1)

# Step 2: Use TF-IDF to convert the combined features into a matrix of TF-IDF␣
↪features
tfidf = TfidfVectorizer(stop_words='english')
tfidf_matrix = tfidf.fit_transform(netflix_df_cleaned['combined_features'])

# Step 3: Compute cosine similarity between all content
cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)

# Display the similarity matrix shape (just for validation)
cosine_sim.shape
```

[53]:
```
(8790, 8790)
```

4 Machine Learning

[55]:
```
# Step 4: Build a Recommendation Function

# Create a function to get recommendations based on cosine similarity
def get_recommendations(title, cosine_sim=cosine_sim, df=netflix_df_cleaned):
    # Get the index of the content that matches the title
    idx = df[df['title'] == title].index[0]

    # Get the pairwise similarity scores of all content with that title
    sim_scores = list(enumerate(cosine_sim[idx]))

    # Sort the content based on the similarity scores
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)

    # Get the scores of the 10 most similar content
    sim_scores = sim_scores[1:11]

    # Get the content indices
    content_indices = [i[0] for i in sim_scores]

    # Return the top 10 most similar content
    return df['title'].iloc[content_indices]

# Test the recommendation function with a sample title
sample_title = "Dick Johnson Is Dead"
recommendations = get_recommendations(sample_title)

recommendations
```

[55]:
```
5795         S.W.A.T.
2785    Triple Threat
5583      Nowhere Boy
2670       Avengement
2026        Honeytrap
1993       The Stolen
4604            Brick
911              Home
4738         Daffedar
2964          Juanita
Name: title, dtype: object
```

[57]:
```python
recommendations = get_recommendations('Dick Johnson Is Dead')
print(recommendations)
```

```
5795         S.W.A.T.
2785    Triple Threat
5583      Nowhere Boy
2670       Avengement
2026        Honeytrap
1993       The Stolen
4604            Brick
911              Home
4738         Daffedar
2964          Juanita
Name: title, dtype: object
```

5 Advanced Genre Visualizations

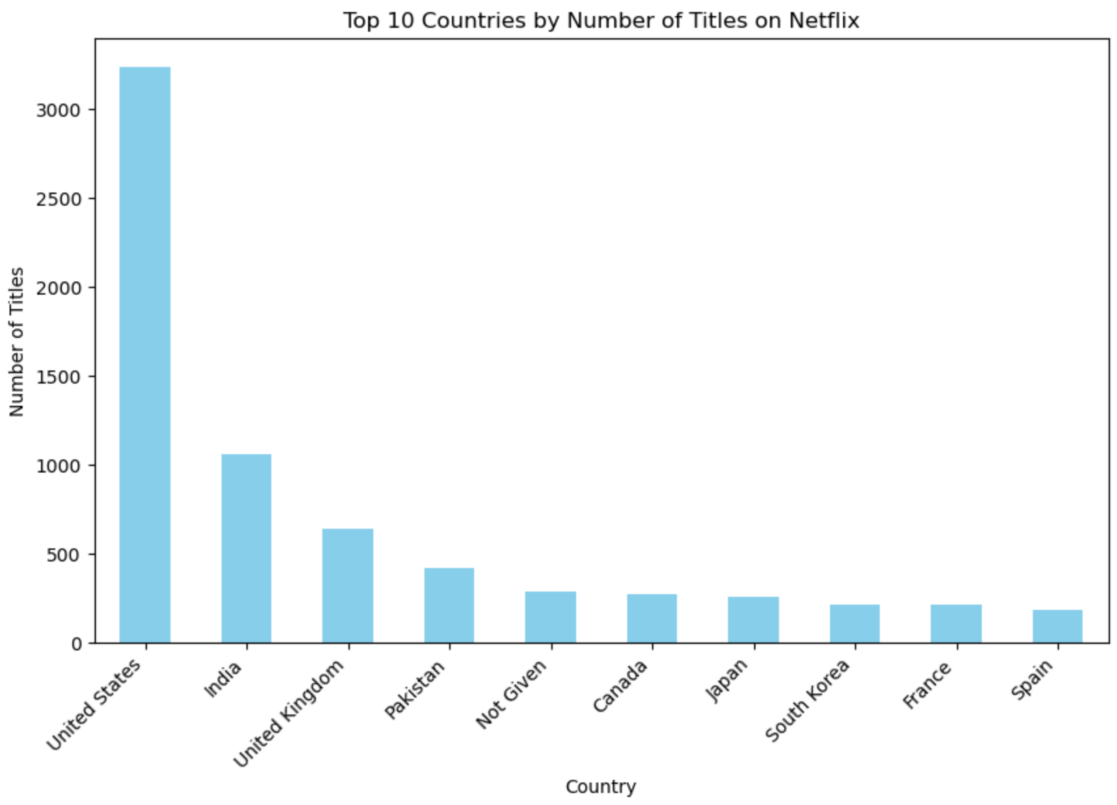
[65]:
```python
# Group by country and count titles
country_distribution = netflix_df_cleaned['country'].value_counts().head(10)
print(country_distribution)

import matplotlib.pyplot as plt

# Plotting the top 10 countries by content count
plt.figure(figsize=(10,6))
country_distribution.plot(kind='bar', color='skyblue')
plt.title('Top 10 Countries by Number of Titles on Netflix')
plt.ylabel('Number of Titles')
plt.xlabel('Country')
plt.xticks(rotation=45, ha='right')
plt.show()
```

```
country
United States     3240
India             1057
United Kingdom     638
```

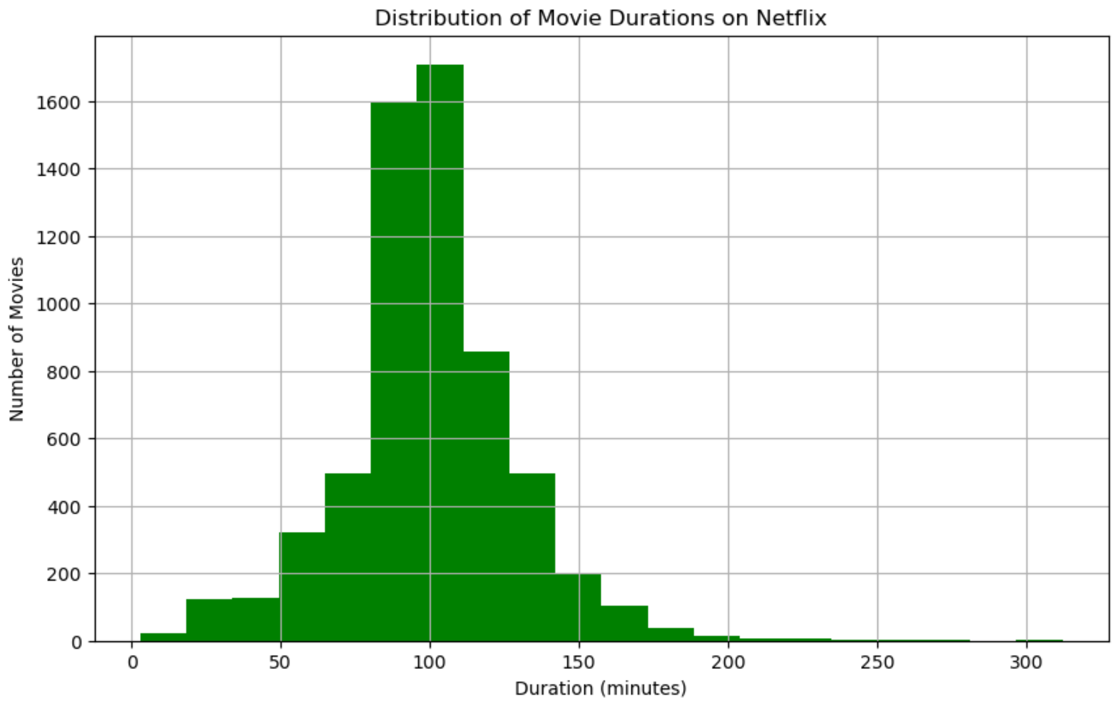
```
Pakistan           421
Not Given          287
Canada             271
Japan              259
South Korea        214
France             213
Spain              182
Name: count, dtype: int64
```



[67]:
```
# Analyze movie duration
movie_duration = netflix_df_cleaned[netflix_df_cleaned['type'] ==␣
↪'Movie']['duration_in_minutes'].describe()
print(movie_duration)

# Plotting the distribution of movie durations
plt.figure(figsize=(10,6))
plt.hist(netflix_df_cleaned[netflix_df_cleaned['type'] ==␣
↪'Movie']['duration_in_minutes'].dropna(), bins=20, color='green')
plt.title('Distribution of Movie Durations on Netflix')
plt.xlabel('Duration (minutes)')
plt.ylabel('Number of Movies')
plt.grid(True)
plt.show()
```

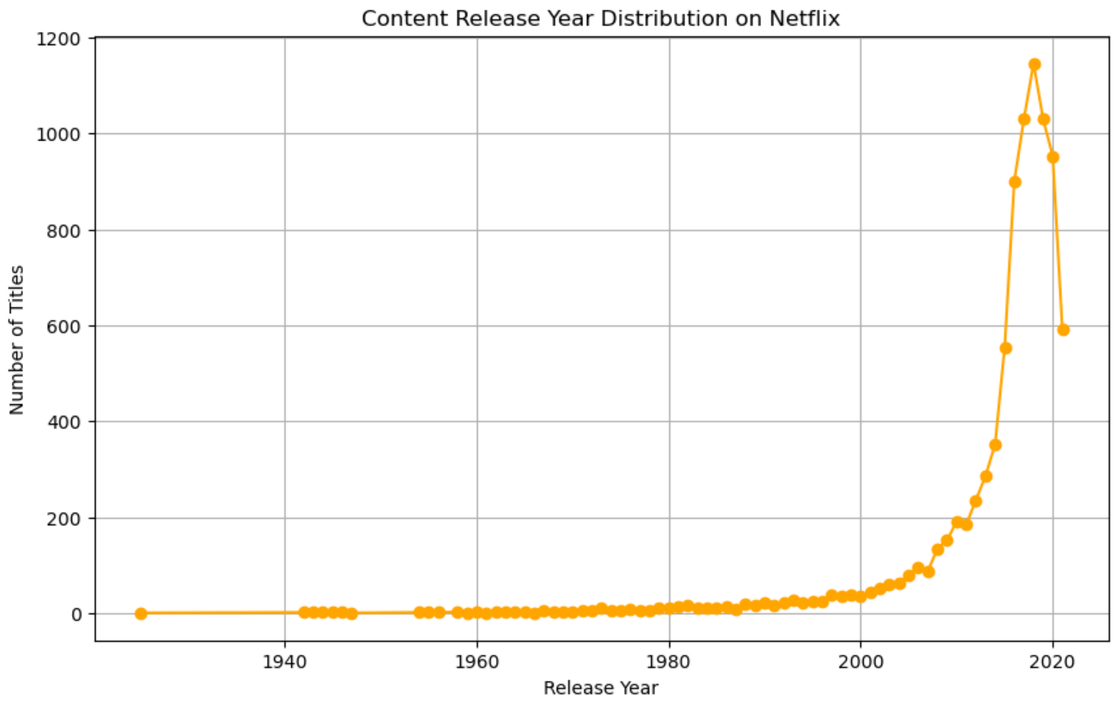
```
count    6126.000000
mean       99.584884
std        28.283225
min         3.000000
25%        87.000000
50%        98.000000
75%       114.000000
max       312.000000
Name: duration_in_minutes, dtype: float64
```



[69]:
```
# Analyze release year distribution
release_year_distribution = netflix_df_cleaned['release_year'].value_counts().
↪sort_index()
print(release_year_distribution.head())  # You can plot this or further analyze␣
↪trends over time

# Plotting the release year distribution
plt.figure(figsize=(10,6))
release_year_distribution.plot(kind='line', marker='o', color='orange')
plt.title('Content Release Year Distribution on Netflix')
plt.ylabel('Number of Titles')
plt.xlabel('Release Year')
plt.grid(True)
plt.show()
```

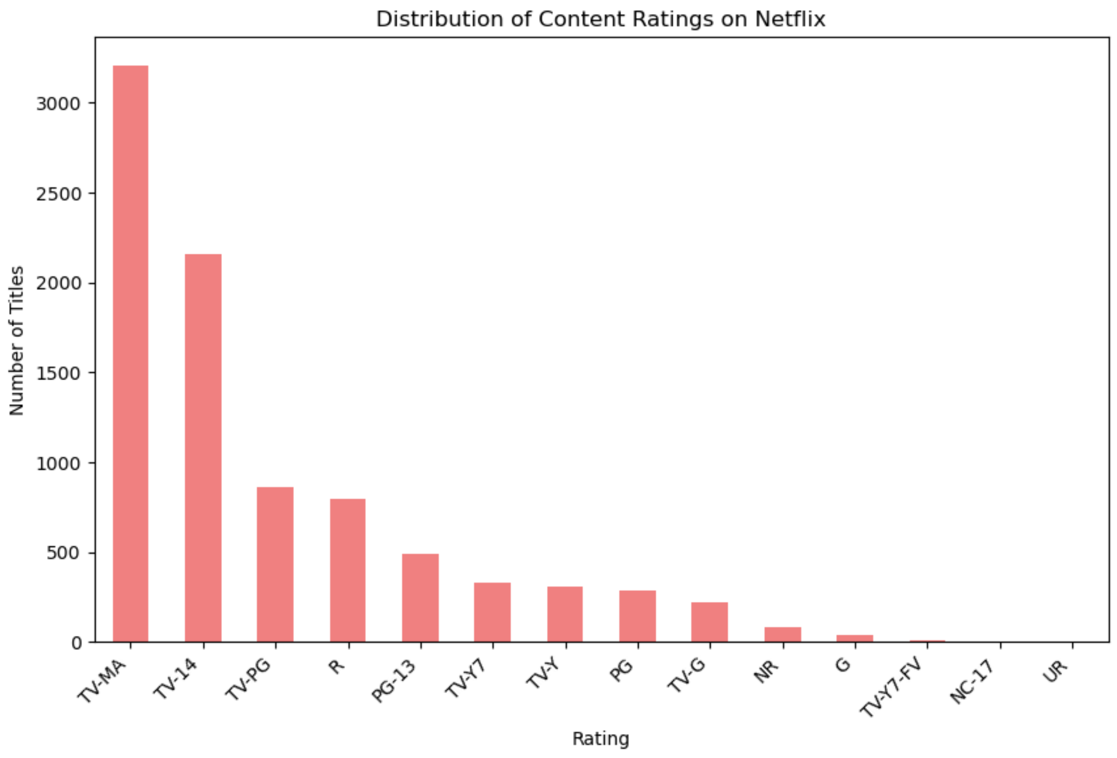
```
release_year
1925    1
1942    2
1943    3
1944    3
1945    4
Name: count, dtype: int64
```



[71]:
```
# Step: Analyze Rating Distribution

# Calculate the distribution of content ratings
rating_distribution = netflix_df_cleaned['rating'].value_counts()

# Plot the distribution of ratings
plt.figure(figsize=(10,6))
rating_distribution.plot(kind='bar', color='lightcoral')
plt.title('Distribution of Content Ratings on Netflix')
plt.ylabel('Number of Titles')
plt.xlabel('Rating')
plt.xticks(rotation=45, ha='right')
plt.show()
```



[73]:
```
# Explode the 'listed_in' column to separate multiple genres
netflix_df_cleaned['genre'] = netflix_df_cleaned['listed_in'].str.split(', ')
netflix_genres_exploded = netflix_df_cleaned.explode('genre')

# Group by country and genre, and count the number of titles
country_genre_distribution = netflix_genres_exploded.groupby(['country',␣
↪'genre']).size().reset_index(name='count')

# Display the top genres for a few countries
print(country_genre_distribution.head())
```

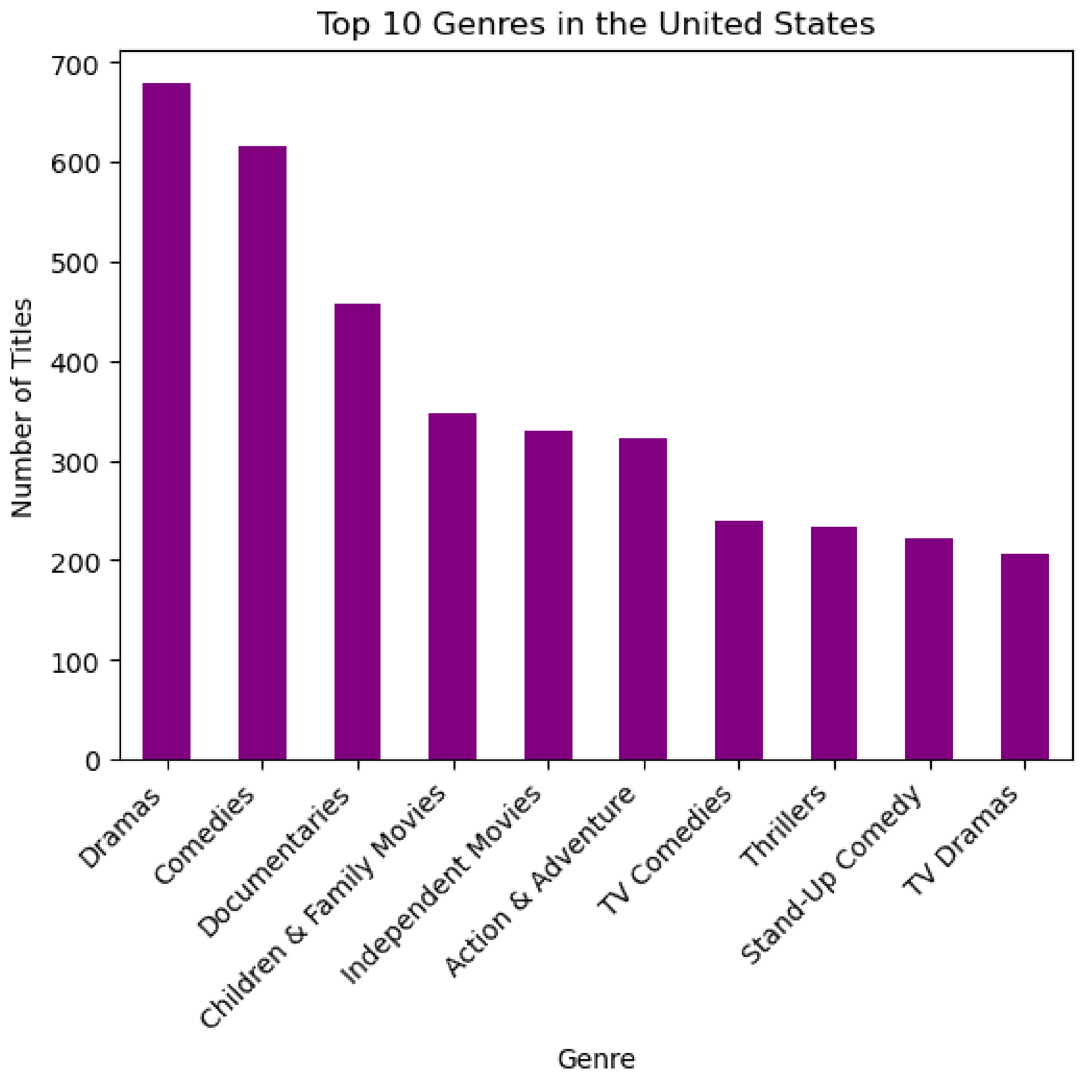
```
     country                   genre  count
0  Argentina      Action & Adventure      2
1  Argentina  Children & Family Movies      2
2  Argentina         Classic & Cult TV      1
3  Argentina          Classic Movies      1
4  Argentina                Comedies     10
```

[75]:
```
# Filter the top genres for a specific country (e.g., 'United States')
```

```
top_genres_us =␣
↪country_genre_distribution[country_genre_distribution['country'] == 'United␣
↪States'].sort_values(by='count', ascending=False).head(10)

# Plot the top genres for the United States
plt.figure(figsize=(10,6))
top_genres_us.plot(kind='bar', x='genre', y='count', color='purple',␣
↪legend=False)
plt.title('Top 10 Genres in the United States')
plt.ylabel('Number of Titles')
plt.xlabel('Genre')
plt.xticks(rotation=45, ha='right')
plt.show()
```

```
<Figure size 1000x600 with 0 Axes>
```

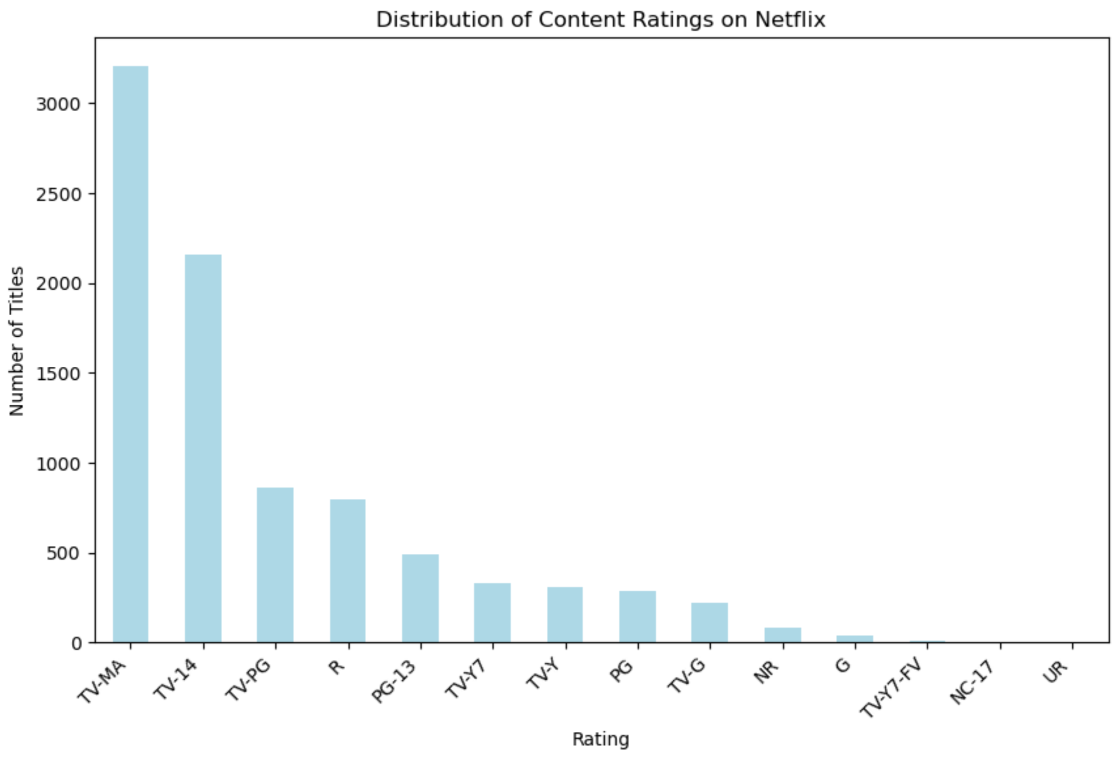


[79]:
```
# Calculate the distribution of content ratings
rating_distribution = netflix_df_cleaned['rating'].value_counts()

# Display the top ratings
print(rating_distribution)

import matplotlib.pyplot as plt# Plot the distribution of ratings
plt.figure(figsize=(10,6))
rating_distribution.plot(kind='bar', color='lightblue')
plt.title('Distribution of Content Ratings on Netflix')
plt.ylabel('Number of Titles')
plt.xlabel('Rating')
plt.xticks(rotation=45, ha='right')
plt.show()
```

```
rating
TV-MA       3205
TV-14       2157
TV-PG        861
R            799
PG-13        490
TV-Y7        333
TV-Y         306
PG           287
TV-G         220
NR            79
G             41
TV-Y7-FV       6
NC-17          3
UR             3
Name: count, dtype: int64
```

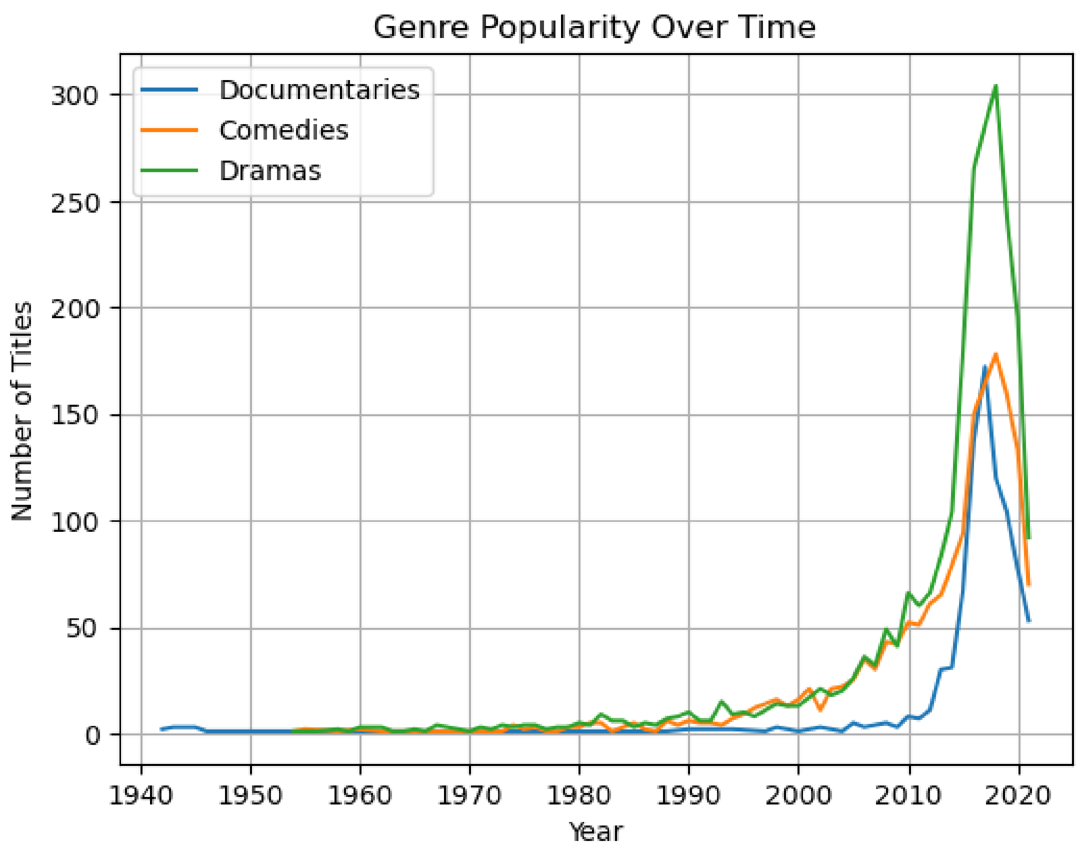


[81]:
```
# Explode the 'listed_in' column to separate genres
netflix_df_cleaned['genre'] = netflix_df_cleaned['listed_in'].str.split(', ')
netflix_genres_exploded = netflix_df_cleaned.explode('genre')

# Group by year and genre to see how many titles are released in each genre per␣
 ↪year
genre_year_distribution = netflix_genres_exploded.groupby(['release_year',␣
 ↪'genre']).size().reset_index(name='count')

# Visualize the popularity of a few selected genres over time
popular_genres = ['Documentaries', 'Comedies', 'Dramas']  # Select a few genres␣
 ↪for visualization
for genre in popular_genres:
    genre_trend = genre_year_distribution[genre_year_distribution['genre'] ==␣
 ↪genre]
    plt.plot(genre_trend['release_year'], genre_trend['count'], label=genre)

plt.title('Genre Popularity Over Time')
plt.xlabel('Year')
plt.ylabel('Number of Titles')
plt.legend()
plt.grid(True)
plt.show()
```



[83]:
```
# Group by release year and genre, and find the top genre for each year
top_genre_per_year = netflix_genres_exploded.groupby(['release_year', 'genre']).
 ↪size().groupby(level=0, group_keys=False).nlargest(1).
 ↪reset_index(name='count')

# Display the top genre for each year
print(top_genre_per_year)
```

```
    release_year                    genre  count
0           1925                 TV Shows      1
1           1942           Classic Movies      2
2           1943            Documentaries      3
3           1944           Classic Movies      3
4           1945           Classic Movies      3
..           ...                      ...    ...
69          2017     International Movies    328
70          2018     International Movies    340
71          2019     International Movies    282
```

```
72          2020    International Movies    239
73          2021  International TV Shows    149

[74 rows x 3 columns]
```

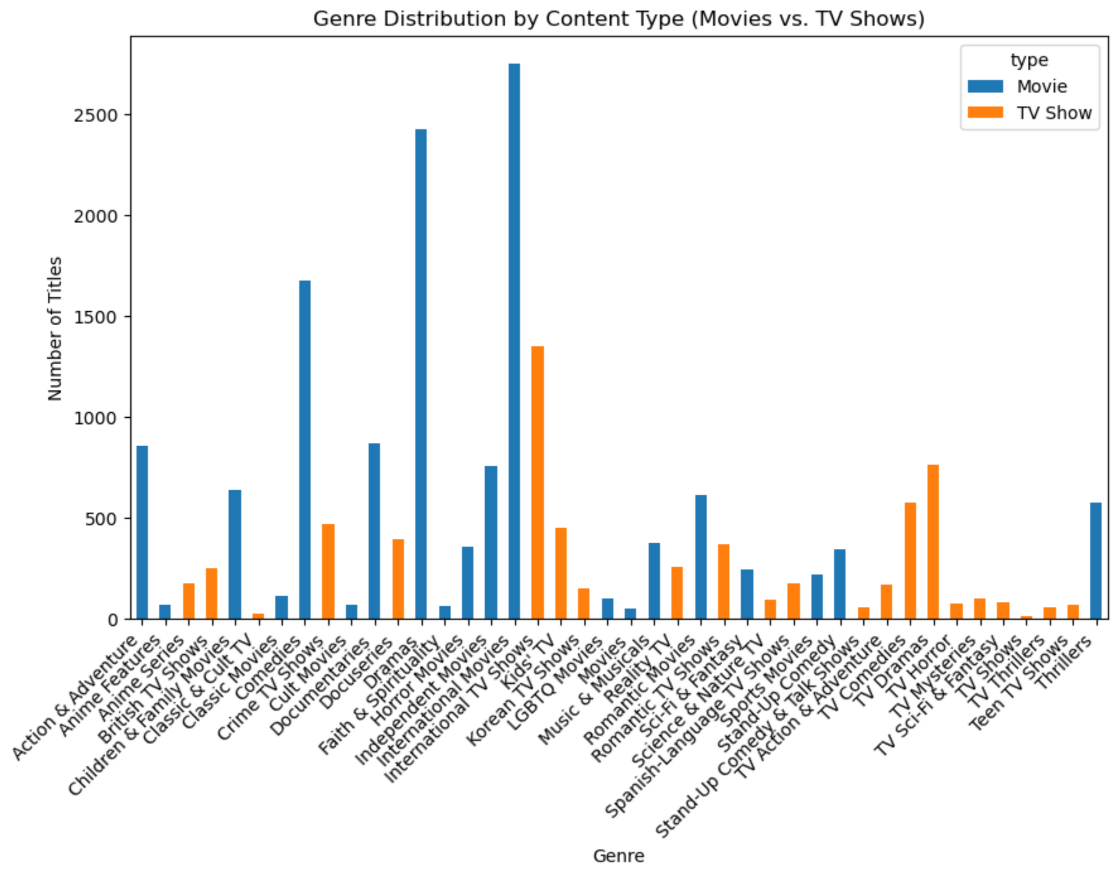
[85]:
```
# Group by genres and count how often different genre combinations appear
genre_combinations = netflix_df_cleaned['listed_in'].value_counts().head(10)

# Display common genre combinations
print(genre_combinations)
```

```
listed_in
Dramas, International Movies                        362
Documentaries                                       359
Stand-Up Comedy                                     334
Comedies, Dramas, International Movies              274
Dramas, Independent Movies, International Movies    252
Kids' TV                                            219
Children & Family Movies                            215
Children & Family Movies, Comedies                  201
Documentaries, International Movies                 186
Dramas, International Movies, Romantic Movies       180
Name: count, dtype: int64
```

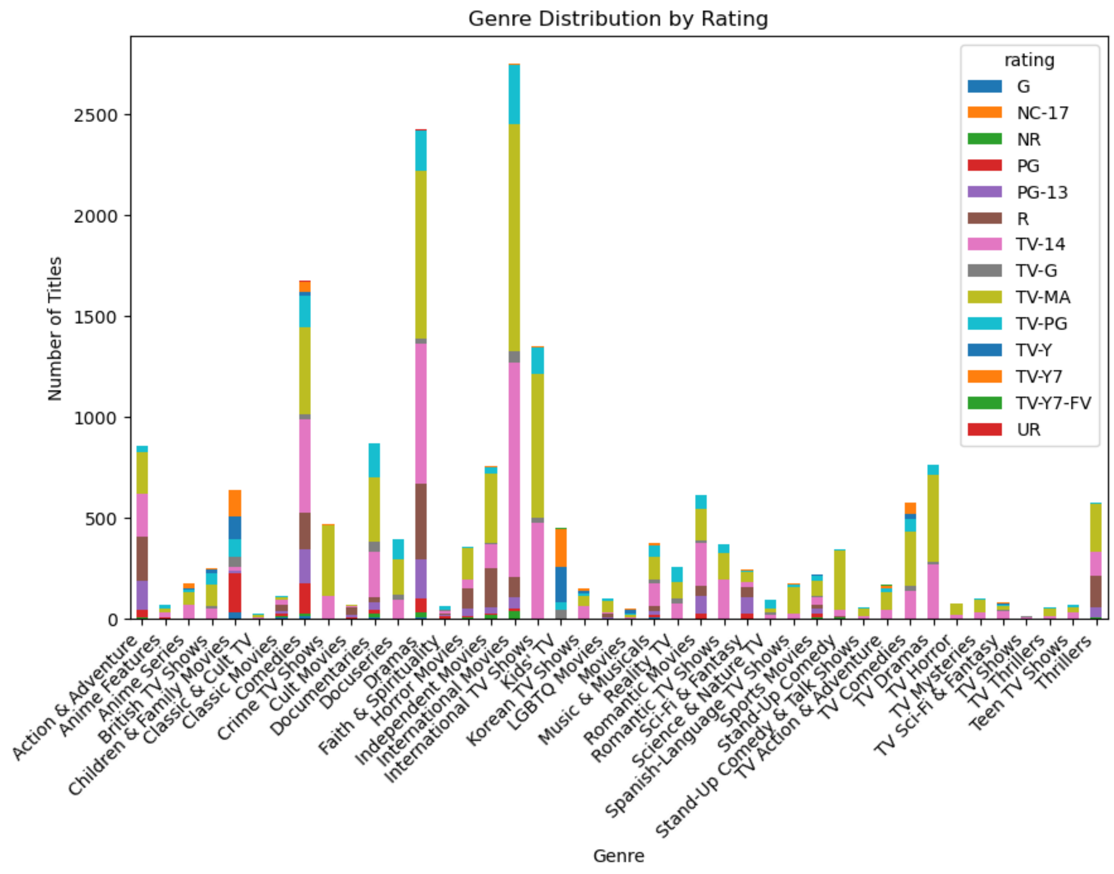
[87]:
```
# Group by content type and genre to compare distribution between Movies and TV␣
↪Shows
genre_type_distribution = netflix_genres_exploded.groupby(['type', 'genre']).
↪size().unstack().fillna(0)

# Plot the distribution for Movies and TV Shows
genre_type_distribution.T.plot(kind='bar', stacked=True, figsize=(10,6))
plt.title('Genre Distribution by Content Type (Movies vs. TV Shows)')
plt.xlabel('Genre')
plt.ylabel('Number of Titles')
plt.xticks(rotation=45, ha='right')
plt.show()
```



[89]:
```
# Group by rating and genre to analyze the distribution
genre_rating_distribution = netflix_genres_exploded.groupby(['rating',␣
↪'genre']).size().unstack().fillna(0)

# Plot the distribution of genres across ratings
genre_rating_distribution.T.plot(kind='bar', stacked=True, figsize=(10,6))
plt.title('Genre Distribution by Rating')
plt.xlabel('Genre')
plt.ylabel('Number of Titles')
plt.xticks(rotation=45, ha='right')
plt.show()
```


Genre Distribution by Rating
Number of Titles
Genre
rating
G
NC-17
NR
PG
PG-13
R
TV-14
TV-G
TV-MA
TV-PG
TV-Y
TV-Y7
TV-Y7-FV
UR
0
500
1000
1500
2000
2500
Action & Adventure
Anime Features
Anime Series
British TV Shows
Children & Family Movies
Classic & Cult TV
Classic Movies
Comedies
Crime TV Shows
Cult Movies
Documentaries
Docuseries
Dramas
Faith & Spirituality
Horror Movies
Independent Movies
International Movies
International TV Shows
Korean TV Shows
Kids' TV
LGBTQ Movies
Movies
Music & Musicals
Reality TV
Romantic Movies
Romantic TV Shows
Sci-Fi & Fantasy
Science & Nature TV
Spanish-Language TV Shows
Sports Movies
Stand-Up Comedy
Stand-Up Comedy & Talk Shows
TV Action & Adventure
TV Comedies
TV Dramas
TV Horror
TV Mysteries
TV Sci-Fi & Fantasy
TV Shows
TV Thrillers
Teen TV Shows
Thrillers

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```