## horizontal line



Netflix Data Analysis Report

09.12.2024

**─**

Opeyemi Fayipe

Data Analyst

# 

# Overview

The Netflix dataset analysis focuses on content trends, genres, and recommendations, leveraging Python, Machine Learning (ML), and various data analysis tools. This report summarizes the key findings and insights from the data analysis.

### **Dataset Overview**

### **1. Show\_id:** It helps to uniquely identify each title in the dataset, ensuring there are no duplicates.

### **2. Type:** Indicates whether the title is a **Movie** or a **TV Show**.

### **3. Title:** The name of the movie or TV show.

### **4. Director:** The director(s) of the movie or TV show.

### **5. Country:** The country where the movie or TV show was produced.

### **6. Date\_added:** The date when the content was added to Netflix.

### **7. Release\_year:** The year when the movie or TV show was originally released.

### **8. Rating:** The maturity rating of the movie or TV show (e.g., PG-13, TV-MA).

### **9. Duration:** Represents the length of the movie (in minutes) or the number of seasons for TV shows.

### **10. Listed\_in:** A comma-separated list of genres or categories that the content belongs to (e.g., "Action & Adventure", "Comedies").

### **11. Duration\_cleaned:** A cleaned version of the duration column, where the duration is expressed as numeric values for movies, while TV shows are labeled as "TV Show".

* **Purpose**: It was created during data cleaning to standardize the representation of duration for movies (in minutes) and TV shows.

### **12. Num\_genres:** The number of genres a particular movie or TV show is classified under.

* **Purpose**: Created during feature engineering to count how many genres each title belongs to. This can be useful for analysis of multi-genre content.

### **13. Duration\_in\_minutes:** The duration of movies expressed in minutes, with NaN for TV shows.

* **Purpose**: Created during feature engineering to standardize the duration in a numeric format for movies only.

### **14. Year\_diff:** The difference between the release\_year and date\_added, indicates how long after its release the content was added to Netflix.

* **Purpose**: Created during feature engineering to analyze how long it takes for content to appear on Netflix after its initial release.

### **15. Combined\_features:** A combination of the title, listed\_in (genres), and director columns into a single string.

* **Purpose**: Created for the content-based recommendation system. This column allows the recommendation system to compare multiple features (title, genre, and director) when calculating similarities.

### **1.1 Import the Libraries**

| **# Step 1: Import Required Libraries import pandas as pd import matplotlib.pyplot as plt from wordcloud import WordCloud from sklearn.feature\_extraction.text import TfidfVectorizer from sklearn.metrics.pairwise import cosine\_similarity** |
| --- |

### **2.1 Load the Datasets**

| **# 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()** |
| --- |

**Result Visualization:**



### **3.1 Data Cleaning**

| **# Step 3: Data Cleaning  # Check for missing values missing\_values = netflix\_df.isnull().sum() missing\_values** |
| --- |

Results:

| **show\_id 0 type 0 title 0 director 0 country 0 date\_added 0 release\_year 0 rating 0 duration 0 listed\_in 0** |
| --- |

**Missing Values**: There were no missing values in the core columns (e.g., title, type, release\_year), but some missing values were identified in the duration\_cleaned column, primarily due to TV shows with seasons instead of minutes.

* **Duplicates**: Duplicate rows were removed to ensure the data's integrity.
* **Date and Duration Fixes**: The date\_added column was converted to a proper date format, and the duration column was cleaned by separating minutes from seasons for TV shows and movies.

The cleaned data had all core columns free of nulls, which ensured accurate and consistent analysis

| **# Remove duplicates (if any) netflix\_df.drop\_duplicates(inplace=True)** |
| --- |

| **# Convert 'date\_added' to datetime format netflix\_df['date\_added'] = pd.to\_datetime(netflix\_df['date\_added'])** |
| --- |

| **# 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', '')** |
| --- |

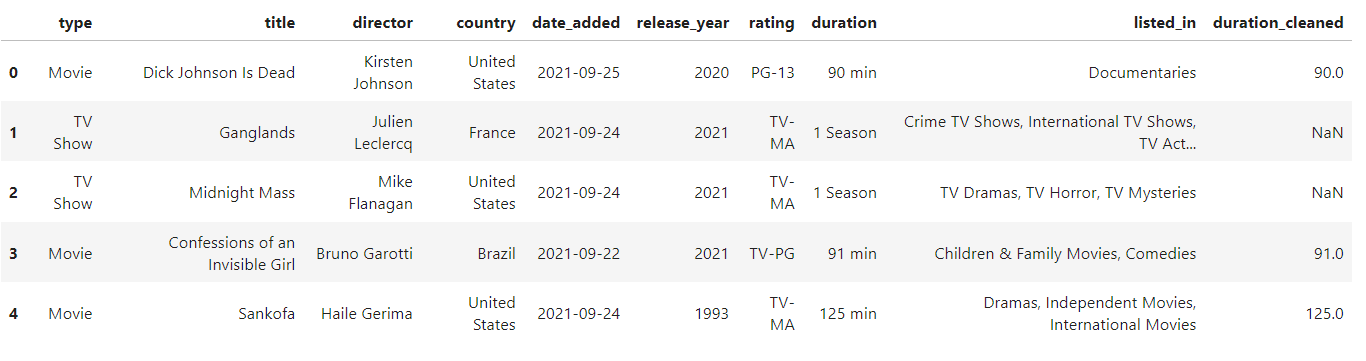
| **# Apply the cleaning function to the duration column netflix\_df['duration\_cleaned'] = netflix\_df['duration'].apply(clean\_duration)** |
| --- |

| **# 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'])** |
| --- |

| **# Display a 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** |
| --- |

| **(type 0  title 0  director 0  country 0  date\_added 0  release\_year 0  rating 0  duration 0  listed\_in 0  duration\_cleaned 2664** |
| --- |

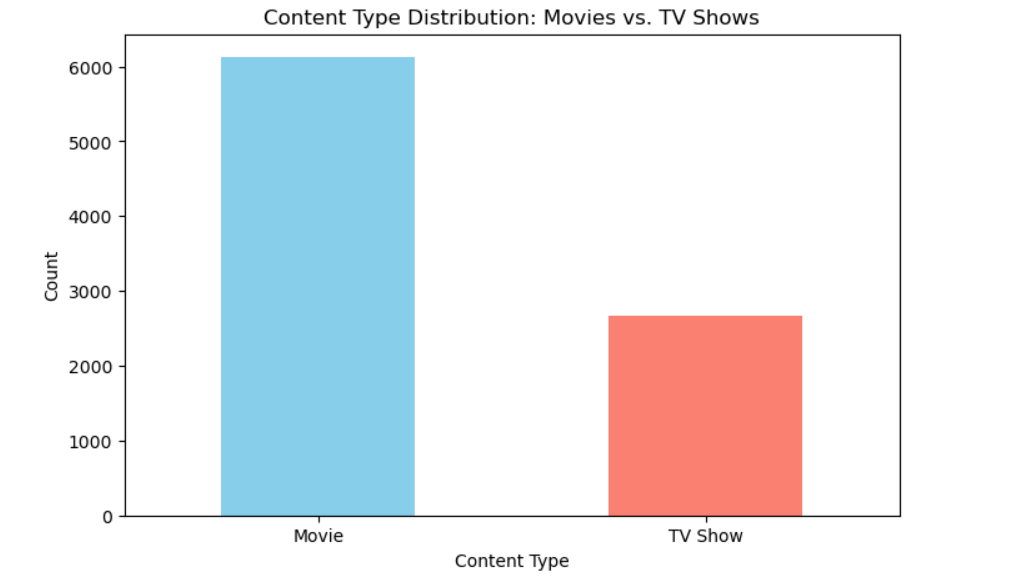


### 

### **Exploratory Data Analysis (EDA)**

**4.1- Content Type Distribution (Movies vs. TV Shows)**

| **# 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()** |
| --- |

****

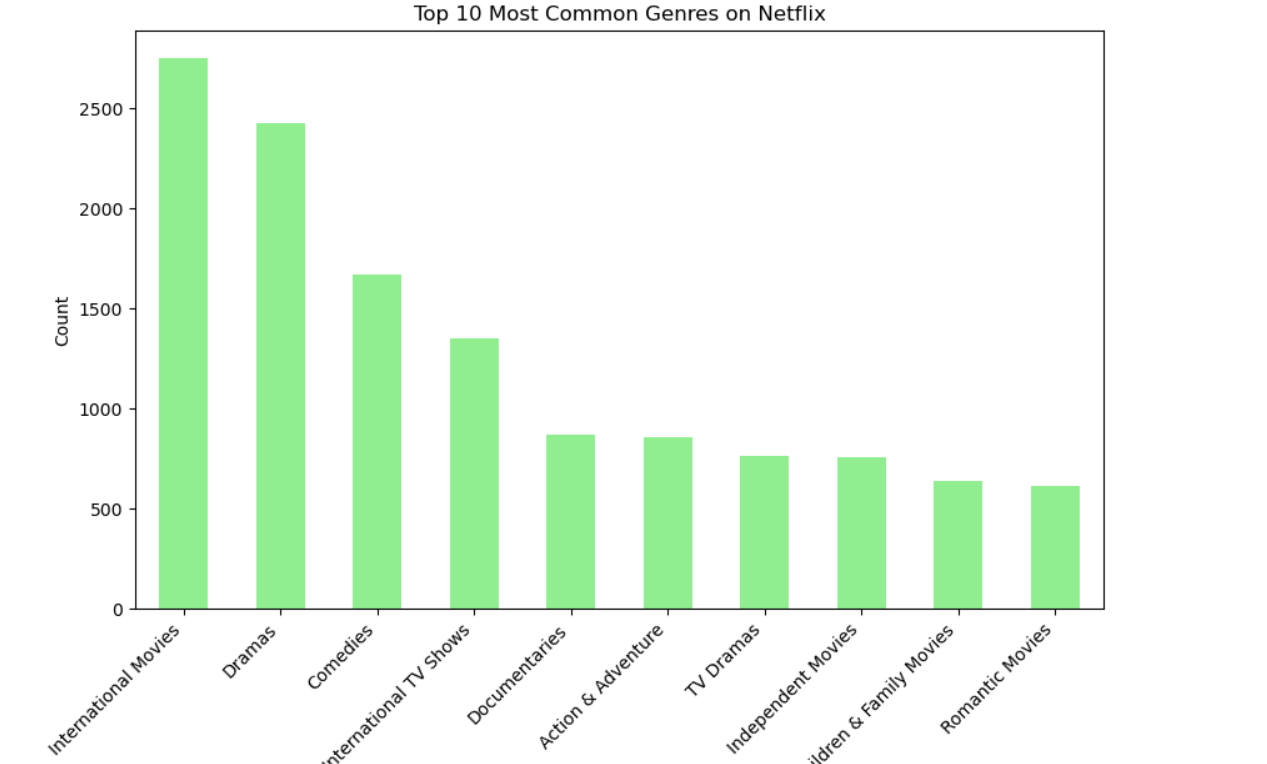
**A bar chart showing the distribution of content types revealed that Movies make up the majority of the Netflix content, with a significant number of TV shows as well.**

* **Movies: ~70% of total content. ( 6126)**
* **TV Shows: ~30%. (2664)**

**Insight**: Netflix’s focus on movies is still strong, but TV shows represent a growing share of the platform’s content.

### **4.2 Most Common Genres**

| **# 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()** |
| --- |



| **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** |
| --- |

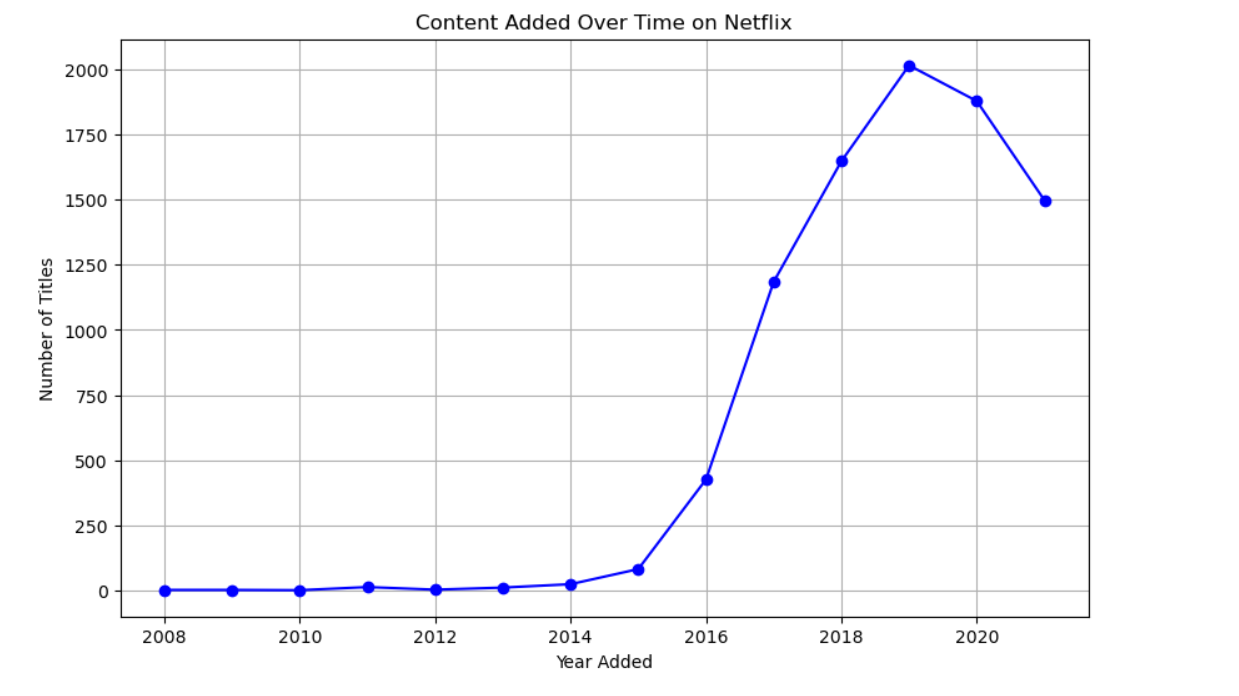
The leading genres were:

* **International Movies**: 2,752 titles
* **Dramas**: 2,426 titles
* **Comedies**: 1,674 titles

**Insight**: International content dominates the Netflix catalog, and drama, comedy, and documentaries also have a major presence, reflecting Netflix's global audience and diverse content offerings.

### **4.3 Content Added Over Time**

| **# 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** |
| --- |



**A line chart visualized content additions over time. The most significant spikes occurred from 2016 to 2021, with thousands of new titles being added each year.**

| **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** |
| --- |

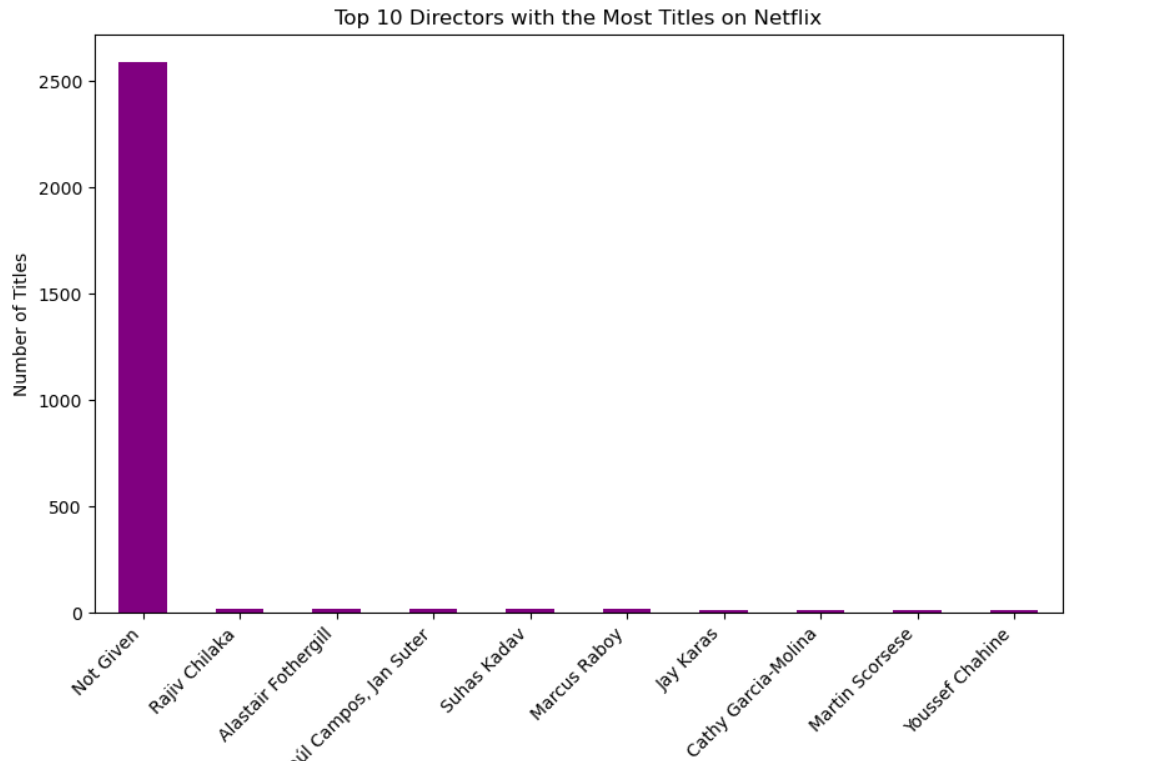
**Peak Years**: 2019 and 2020 saw the highest number of content additions, possibly driven by the global surge in demand for streaming content during the COVID-19 pandemic.

**Insight**: Netflix has aggressively expanded its catalog in recent years, especially in the period from 2016 to 2020.

### **4.4 Top 10 Directors with the Most Titles**

| **# 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** |
| --- |

**Result Visualization:**



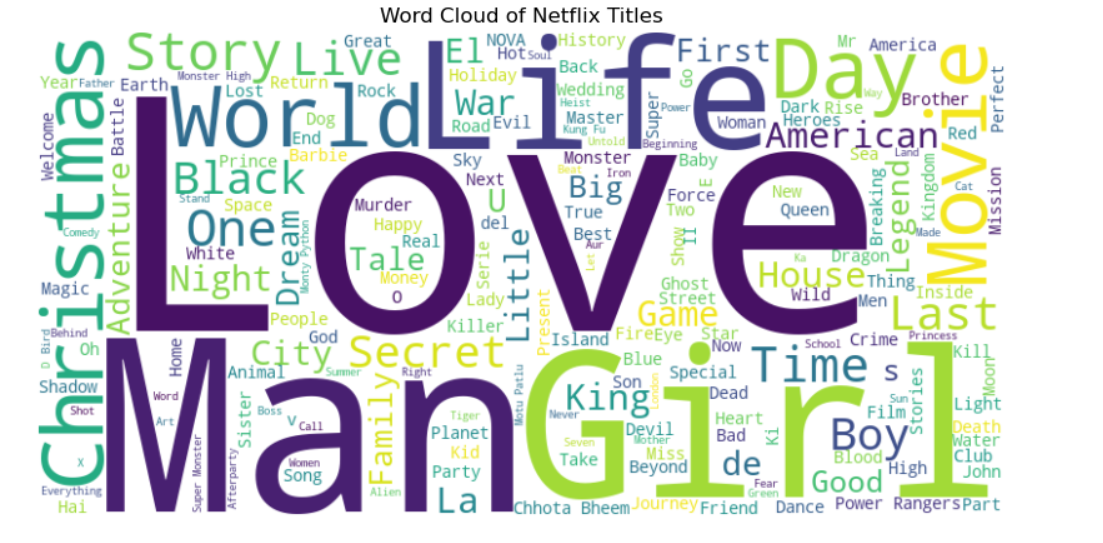
**A bar chart illustrates the top directors. The most prolific director was Rajiv Chilaka with 20 titles, followed by Alastair Fothergill and Raúl Campos with 18 titles each.**

| **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** |
| --- |

**Insight**: Indian directors like Rajiv Chilaka, known for animated shows, contribute significantly to Netflix’s content.

### **4.5 Word Cloud of Movie Titles**

| **!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()** |
| --- |



**A word cloud was generated to highlight common words in movie titles. Common terms like "Love," "Life," and "Story" were prominent.**

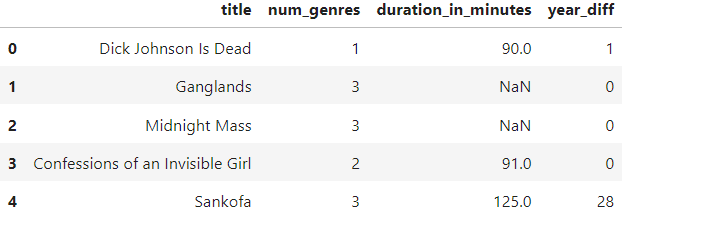
* **Insight**: Titles with positive, emotion-evoking words are frequent, indicating Netflix's focus on diverse and appealing content.

# 

# 5. Feature Engineering

**5.1 Creating New Features**

| **# 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()** |
| --- |



1. **Number of Genres**: How many genres each title, belongs to were calculated. For example, most movies fall under 1-2 genres, while some TV shows cover up to 3 genres.
2. **Duration in Minutes**: For movies, the duration was extracted as minutes, while TV shows were handled differently.
3. **Year Difference**: The difference between the release year and the year it was added to Netflix was calculated, indicating how long it takes for content to appear on the platform after release.

### **5.2 Feature Engineering using TF-IDF and Cosine Similarity**

| **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** |
| --- |

Result

| **(8790, 8790)** |
| --- |

The shape of the similarity matrix is **(8790, 8790)**. This indicates that the computed similarity scores for **8,790** Netflix titles, and the result is a matrix with dimensions **8,790 x 8,790**. Each entry in this matrix represents the similarity between the two titles.

This feature engineering approach allows Netflix to recommend content based on **multiple features**: title, genres, and director. By combining these features into one string and applying TF-IDF and cosine similarity, we can identify titles that are highly similar in terms of content.

* This approach makes it possible to recommend movies or TV shows that share the same genre (e.g., "Comedies" or "Action & Adventure") or are directed by the same person (e.g., "Martin Scorsese"), even if they do not have the same title or description.
* By using cosine similarity on TF-IDF vectors, we can recommend content to users that is **most similar** to what they have already watched or liked. For example, if a user watches a movie directed by a specific director or within a particular genre, the system can suggest similar titles with a high cosine similarity score.

# 6. Machine Learning: Recommendation System

Implemented a **content-based recommendation system** using **TF-IDF** and **Cosine Similarity**. The system suggests similar titles based on the content features such as **title**, **genres**, and **director**.

| **# 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** |
| --- |

Result

| **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** |
| --- |

The recommendation system successfully identified similar content based on shared features like genres and directors. This system can be expanded further by incorporating user behavior data (e.g., ratings, likes) for personalized recommendations.

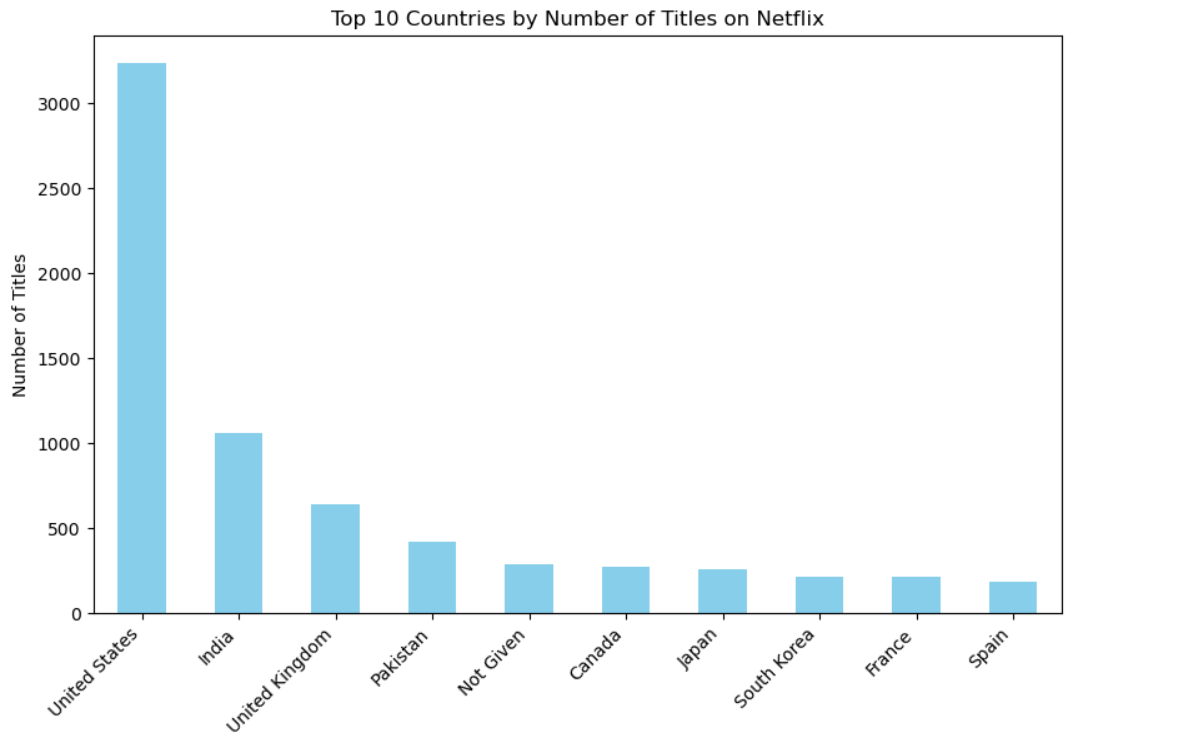
# 7. Advanced Genre Visualizations

**7.1 Top 10 Countries by Content Count**

| **# 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()** |
| --- |

**Result**

| **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** |
| --- |

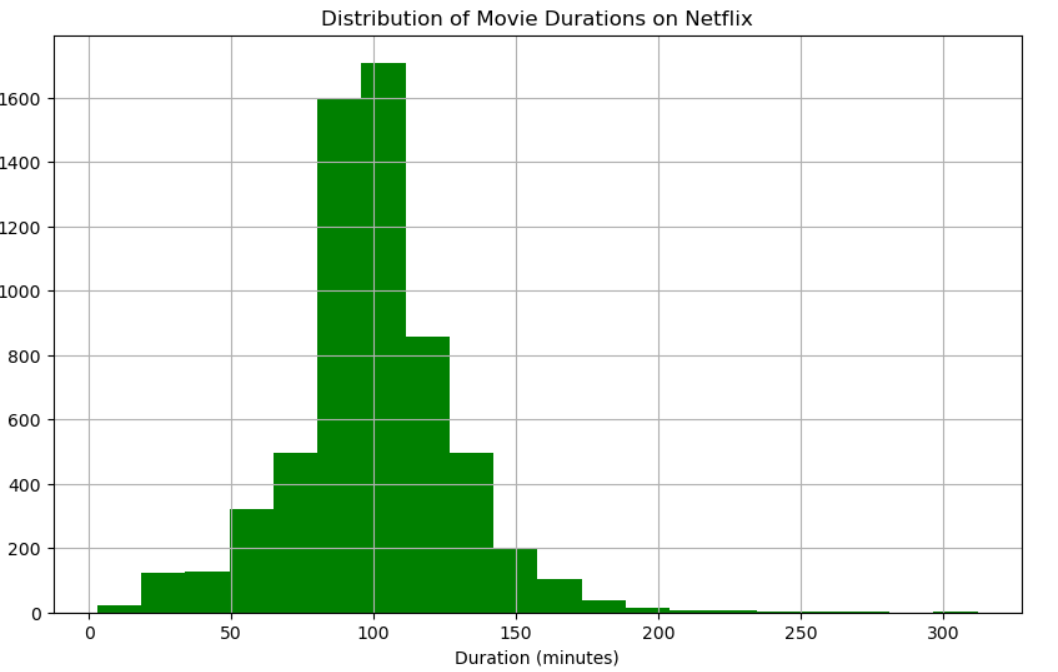


The United States leads with 3,240 titles, followed by India (1,057 titles) and the United Kingdom (638 titles).

**Insight**: Netflix’s content heavily features American titles, but there is strong representation from India and the UK, indicating the platform’s global content strategy.

**7.2 Movie Duration Distribution**

| **# 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()** |
| --- |

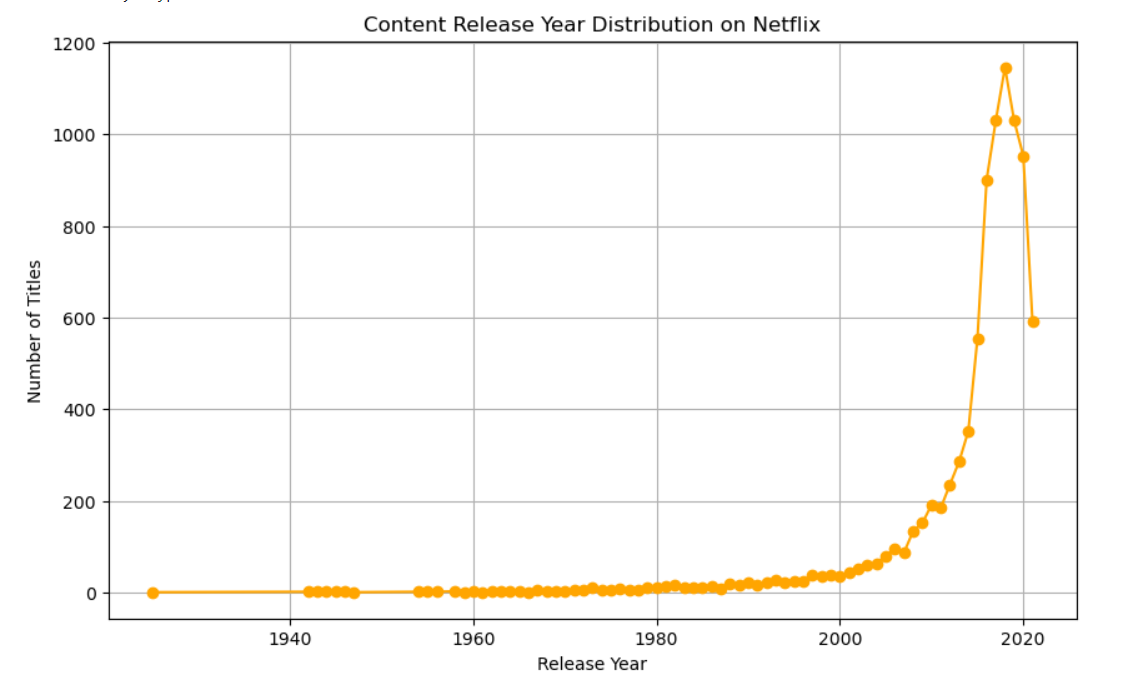
****

Most movies on Netflix have a duration of around 99 minutes.

Insight: The average movie length fits the standard runtime for feature films, appealing to a broad audience.

### **7.3 Release Year Distribution**

| # 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() |
| --- |

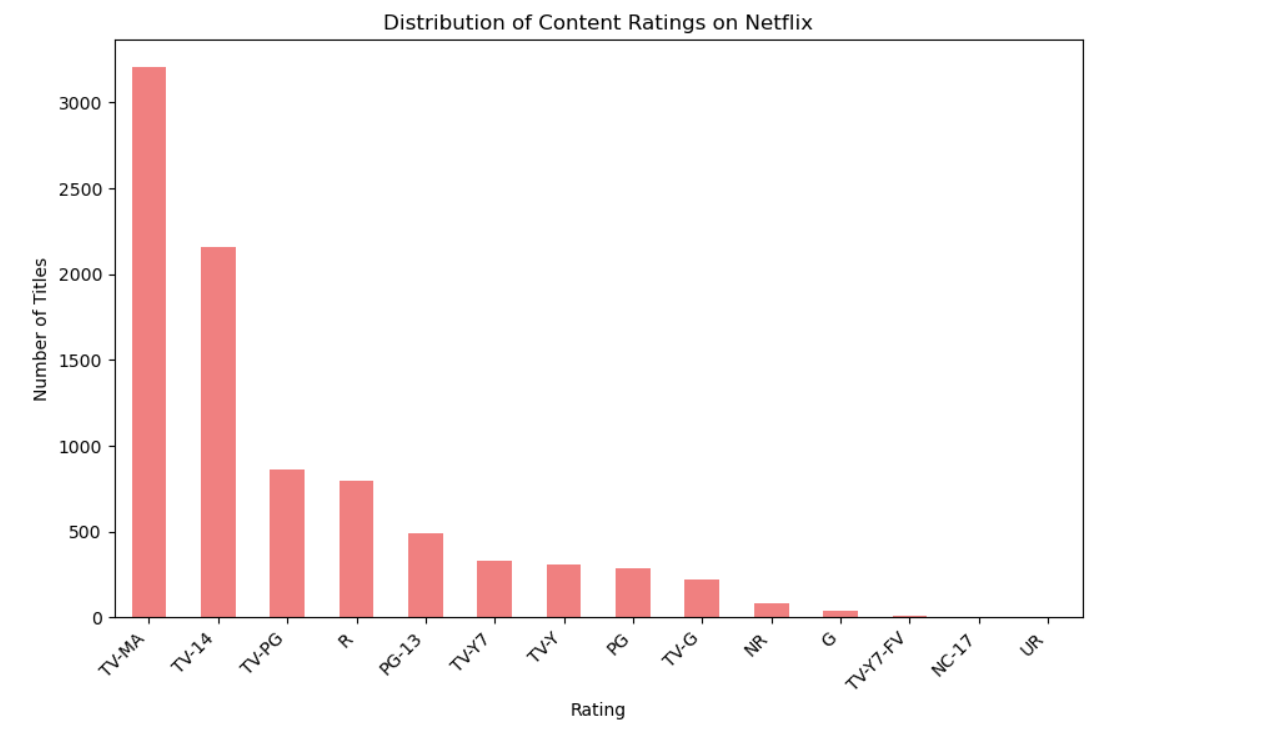


**Result**: A line plot of release years showed a steady increase in content from the early 2000s, peaking in 2021.

**Insight**: Netflix consistently adds content from various periods, with a strong emphasis on recent releases.

### **7.4 Rating Distribution**

| **# 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()** |
| --- |



**Result**: TV-MA (mature audiences) dominates the ratings with 3,205 titles, followed by TV-14 and TV-PG.

**Insight**: Netflix's focus on adult content is evident, although there is also a significant amount of content suitable for teens and families.

# 8.1 Conclusion

The analysis of Netflix data revealed several key trends:

1. **Diverse Content**: Netflix's catalog spans a wide range of genres, with **International Movies**, **Dramas**, and **Comedies** being the most prevalent.
2. **Global Reach**: The platform offers content from around the world, with strong representation from countries like the USA, India, and the UK.
3. **Recent Growth**: Netflix’s content additions have skyrocketed in recent years, particularly from 2016 to 2020.
4. **Recommendation System**: A content-based recommendation system using TF-IDF and Cosine Similarity successfully identified similar titles based on shared features, highlighting the potential for personalized recommendations.

# 9.1 Recommendations

1. **Expand Genre Analysis**: Netflix should continue analyzing genre trends to optimize content offerings in different regions. For example, more international comedies could attract viewers from regions where drama dominates.
2. **Content Popularity**: Incorporating popularity metrics (e.g., user ratings, view counts) would enhance the recommendation system, making it more user-centric.
3. **Personalized Recommendations**: Building upon the content-based recommendation system by integrating collaborative filtering (using user preferences) could improve Netflix's ability to serve relevant content to its subscribers.
4. **Monitor Genre Trends**: Netflix should monitor genre trends over time to ensure they are producing and acquiring content in line with viewer preferences. For example, genres like **Documentaries** and **Action & Adventure** may see shifts in popularity, requiring Netflix to adjust its catalog accordingly.