Netflix Business Case - Avinash Patil

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
!gdown 1b0nz3Iy1rNN-6Hmpfbq0_V6WPwNzckAN
df = pd.read_csv('netflix.csv')
df.head()
```

```
Downloading...
From: https://drive.google.com/uc?id=1b0nz3Iy1rNN-6Hmpfbq0_V6WPwNzckAN
To: /content/netflix.csv
100% 3.40M/3.40M [00:00<00:00, 37.1MB/s]
```

	show_id	type	title	director	cast	country	date_added	release_
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	

1. Defining Problem Statement and Analysing basic metrics (10 Points)

Netflix aims to identify data-driven insights to decide the types of shows or movies to produce and strategize how to grow their business in various countries. This includes:

1. Understanding the current types of content available.
2. Exploring trends in movies and TV shows across time.
3. Investigating country-specific content preferences.
4. Identifying key actors, directors, and genres driving engagement.

The goal is to provide actionable recommendations to Netflix executives for content production and market expansion

Basic Metrics Analysis We'll analyze:

1. Total Entries: How many records are in the dataset.
2. Content Types: How many entries are Movies vs. TV Shows.
3. Release Year: Distribution of release years.
4. Country Availability: Number of unique countries.
5. Genre Diversity: Popular genres.
6. Date Added: Recent vs. older content trends.

```
total_entries = df.shape[0]
print(f"Total Entries: {total_entries}")
```

```
Total Entries: 8807
```

```
content_types = df['type'].value_counts()
print("Content Types:")
print(content_types)
```

```
Content Types:
type
Movie      6131
TV Show    2676
Name: count, dtype: int64
```

```
release_years = df['release_year'].value_counts().sort_index()
print("Release Years:")
print(release_years)
```

```
Release Years:
release_year
1925       1
1942       2
1943       3
1944       3
1945       4
        ...
2017    1032
2018    1147
2019    1030
2020     953
2021     592
Name: count, Length: 74, dtype: int64
```

```
unique_countries = df['country'].nunique()
print(f"Unique Countries: {unique_countries}")
```

```
Unique Countries: 748
```

```
top_genres = df['listed_in'].str.split(', ', expand=True).stack().value_counts()
print("Top Genres:")
print(top_genres)
```

```
Top Genres:
International Movies      2752
Dramas                    2427
Comedies                  1674
International TV Shows    1351
Documentaries              869
Name: count, dtype: int64
```

```
df['date_added_sorted'] = pd.to_datetime(df['date_added'], errors='coerce')
date_added_counts = df['date_added_sorted'].dropna().sort_values(ascending=False
print("Date Added:")
print(date_added_counts)
```

```
Date Added:
0      2021-09-25
6      2021-09-24
10     2021-09-24
9      2021-09-24
8      2021-09-24
          ...
7370   2010-11-01
5955   2009-11-18
5956   2009-05-05
6611   2008-02-04
5957   2008-01-01
Name: date_added_sorted, Length: 8709, dtype: datetime64[ns]
```

Basic Metrics Analysis Results

Total Entries: The dataset contains 8,807 entries, representing a mix of Movies and TV Shows.

Content Types: Movies: 6,131 entries (69.6%). TV Shows: 2,676 entries (30.4%).

Unique Release Years: The content spans release years from 1925 to 2021, showing a broad historical range of movies and TV shows.

Unique Countries: Content is produced in 748 unique countries, indicating diverse production origins.

Top Genres: The most common genres are:

International Movies - 2752

Dramas - 2427

Comedies - 1674

International TV Shows -1351

Documentaries - 869

Date Added: Recent entries include content added as recently as September 2021. For example: "September 25, 2021" "September 24, 2021" (multiple entries).

2. Observations on the shape of data, data types of all the attributes, conversion of categorical attributes to 'category' (If required), missing value detection, statistical summary (10 Points)

Step 1: Shape of the Data

The shape tells us the number of rows and columns in the dataset.

Rows - 8807

columns - 12

```
data_shape = df.shape
data_shape
```

```
(8807, 13)
```

Step 2: Data Types of All Attributes

Each column in the dataset has a specific data type (e.g., object, int64, float64). We'll list the data types to identify categorical and numerical attributes.

```
data_types = df.dtypes
data_types
```

	0
show_id	object
type	object
title	object
director	object
cast	object
country	object
date_added	object
release_year	int64
rating	object
duration	object
listed_in	object
description	object
date_added_sorted	datetime64[ns]

dtype: object

Step 3: Conversion of Categorical Attributes to 'Category'

Categorical attributes can be optimized by converting them to the category data type. This reduces memory usage and improves performance. We'll identify suitable columns and convert them.

```
df.select_dtypes(include=['category']).columns
```

```
Index([], dtype='object')
```

```
for col in df.select_dtypes(include=['object']).columns:
    print(f"{col}: {df[col].nunique()}")
```

```
show_id: 8807
type: 2
title: 8807
director: 4528
cast: 7692
country: 748
date_added: 1767
rating: 17
duration: 220
listed_in: 514
description: 8775
```

```
memory_before = df.memory_usage(deep=True).sum()
memory_before /= 1024 ** 2  # Convert to MB
print(f"Memory Usage Before: {memory_before:.2f} MB")
```

```
Memory Usage Before: 8.99 MB
```

```
df['type'] = df['type'].astype('category')
df['country'] = df['country'].astype('category')
```

```
memory_after = df.memory_usage(deep=True).sum()
memory_after /= 1024 ** 2  # Convert to MB
print(f"Memory Usage After: {memory_after:.2f} MB")
```

```
Memory Usage After: 8.01 MB
```

Note

As you can see that there are only two columns where it makes more sense to convert them to category since they have very to minimal unique values in them. Hence memory usage is reduced to very little which is still a better option when it comes to dealing with even bigger dataset.

Step 4:

Missing Value Detection

We'll identify columns with missing values and calculate their percentages to decide whether to handle or ignore them.

```
missing_values = df.isnull().sum()
missing_values
```

	0
show_id	0
type	0
title	0
director	2634
cast	825
country	831
date_added	10
release_year	0
rating	4
duration	3
listed_in	0
description	0
date_added_sorted	98

dtype: int64

```
missing_percent = (missing_values / len(df)) * 100
missing_percent
```

	0
show_id	0.000000
type	0.000000
title	0.000000
director	29.908028
cast	9.367549
country	9.435676
date_added	0.113546
release_year	0.000000
rating	0.045418
duration	0.034064
listed_in	0.000000
description	0.000000
date_added_sorted	1.112751

dtype: float64

Step 5: Statistical Summary

For numerical attributes, we'll provide a statistical summary (mean, median, std, min, max, etc.). For categorical attributes, we'll provide the count of unique values.

```
numerical_summary = df.describe()
numerical_summary
```

	release_year	date_added_sorted
count	8807.000000	8709
mean	2014.180198	2019-05-23 01:45:29.452290816
min	1925.000000	2008-01-01 00:00:00
25%	2013.000000	2018-04-20 00:00:00
50%	2017.000000	2019-07-12 00:00:00
75%	2019.000000	2020-08-26 00:00:00
max	2021.000000	2021-09-25 00:00:00
std	8.819312	NaN

```
categorical_summary = df.describe(include=['category'])
categorical_summary
```

	type	country
count	8807	7976
unique	2	748
top	Movie	United States
freq	6131	2818

3. Non-Graphical Analysis: Value counts and unique attributes (10 Points)

For Non-Graphical Analysis, focusing on value counts and unique attributes means identifying how often different values appear in categorical columns and summarizing the uniqueness of the data

1. Unique Attributes

```
unique_attributes = df.nunique()
print("Unique Attributes for each column:\n")
print(unique_attributes)
```

```
Unique Attributes for each column:

show_id              8807
type                    2
title                8807
director             4528
cast                 7692
country               748
date_added           1767
release_year           74
rating                 17
duration              220
listed_in             514
description          8775
date_added_sorted    1699
dtype: int64
```

2. Value Counts

```
for column in df.select_dtypes(include=['object', 'category']).columns:
    print(f"\nValue Counts for column: {column}")
    print(df[column].value_counts())
```

```

Value Counts for column: show_id
show_id
s1       1
s5875    1
s5869    1
s5870    1
s5871    1
        ..
s2931    1
s2930    1
s2929    1
s2928    1
s8807    1
Name: count, Length: 8807, dtype: int64

Value Counts for column: type
type
Movie      6131
TV Show    2676
Name: count, dtype: int64

Value Counts for column: title
```

```
title
Dick Johnson Is Dead                        1
Ip Man 2                                    1
Hannibal Buress: Comedy Camisado            1
Turbo FAST                                  1
Masha's Tales                               1
                                           ..
Love for Sale 2                             1
ROAD TO ROMA                                1
Good Time                                   1
Captain Underpants Epic Choice-o-Rama       1
Zubaan                                      1
Name: count, Length: 8807, dtype: int64

Value Counts for column: director
director
Rajiv Chilaka                     19
Raúl Campos, Jan Suter            18
Marcus Raboy                      16
Suhas Kadav                       16
Jay Karas                         14
                                  ..
Raymie Muzquiz, Stu Livingston     1
Joe Menendez                       1
Eric Bross                         1
Will Eisenberg                     1
Mozez Singh                        1
Name: count, Length: 4528, dtype: int64

Value Counts for column: cast
cast
David Attenborough
Vatsal Dubey, Julie Tejwani, Rupa Bhimani, Jigna Bhardwaj, Rajesh Kava, Mou
Samuel West
Jeff Dunham
David Spade, London Hughes, Fortune Feimster
```

-> Insights to focus on -

1. Most Common Categories: Identify the most frequent values in columns like type, country, and rating.
2. Rare Categories: Pay attention to values with very low frequencies. They may indicate unique or niche data points.
3. Highly Unique Columns: Columns like show_id, title, and description have a high number of unique values, which could make them less relevant for frequency-based analysis.

4. Visual Analysis - Univariate, Bivariate after pre-processing of the data

Note: Pre-processing involves unnesting of the data in columns like Actor, Director, Country

4.1 For continuous variable(s): Distplot, countplot, histogram for univariate analysis (10 Points)

4.2 For categorical variable(s): Boxplot (10 Points)

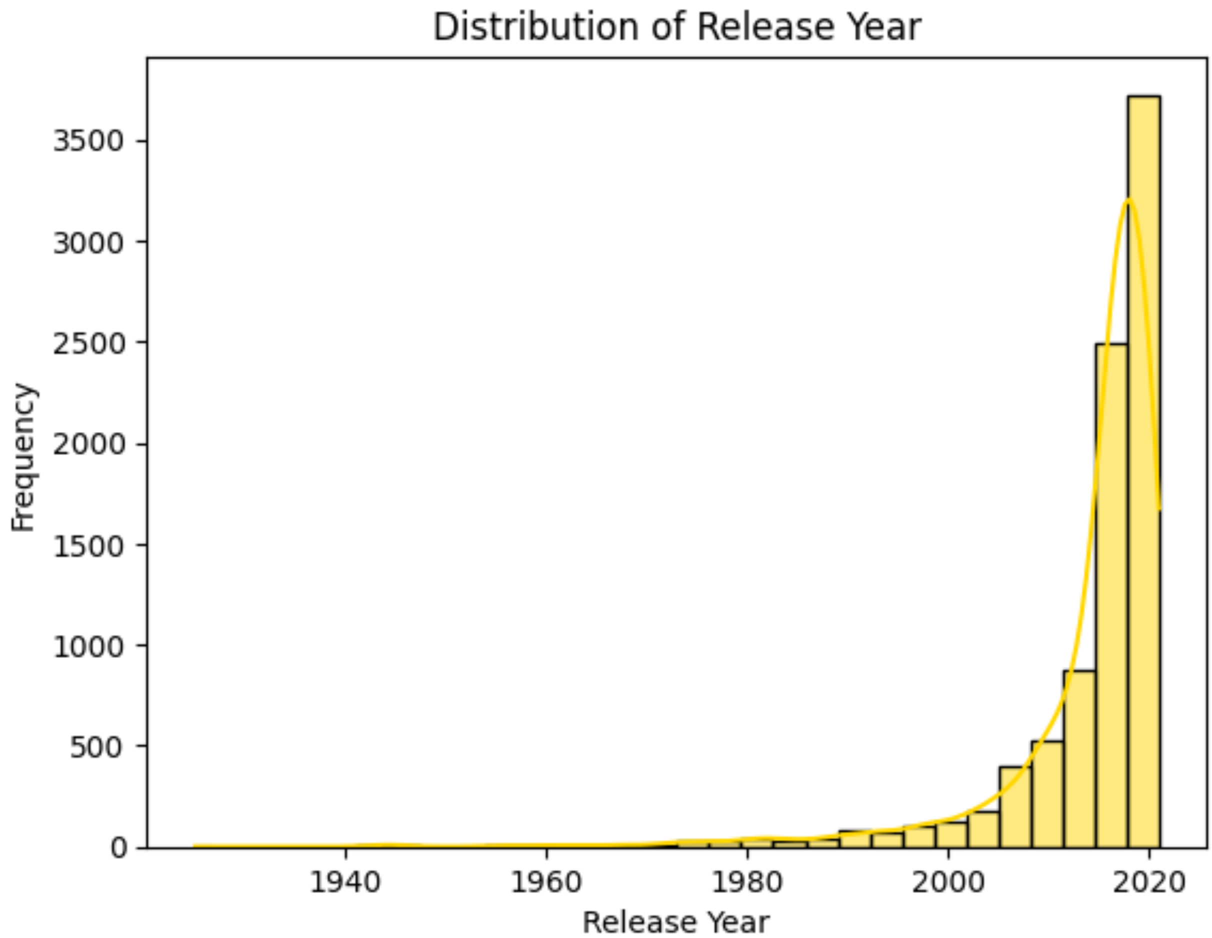
4.3 For correlation: Heatmaps, Pairplots (10 Points)

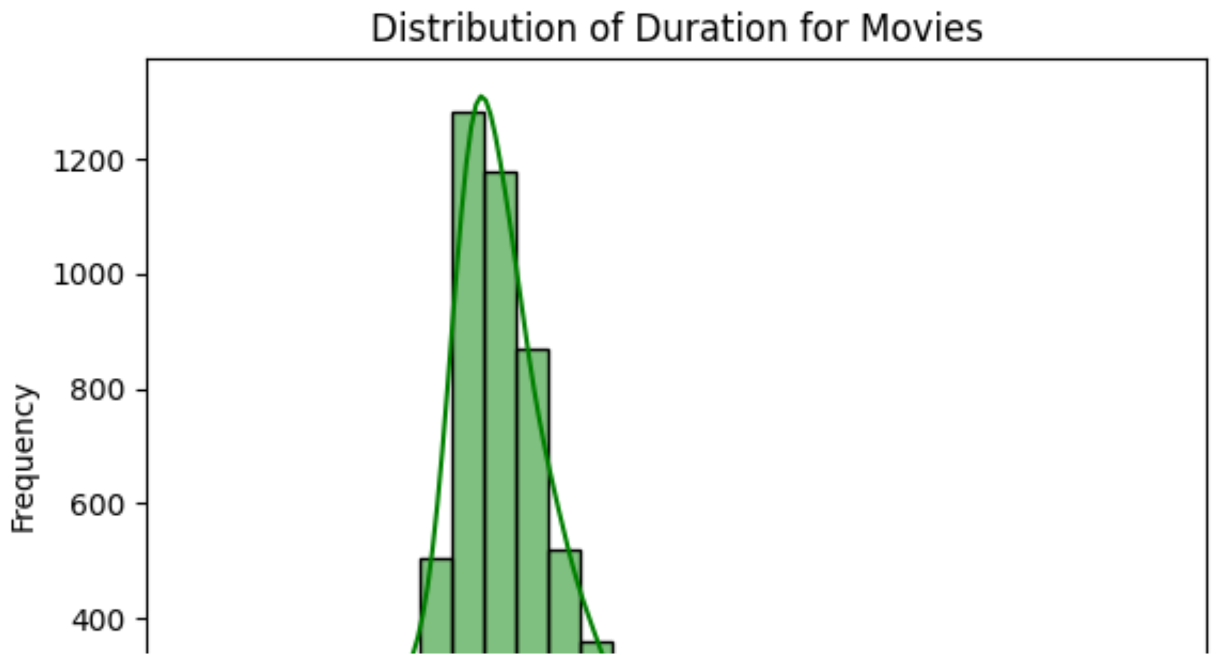
```
df_movies = df[df['type'] == 'Movie']
df_tv_shows = df[df['type'] == 'TV Show']

df_movies['numerical_duration1'] = df_movies['duration'].str.extract('(\d+)').as
df_tv_shows['numerical_duration1'] = df_tv_shows['duration'].str.extract('(\d+)'
```

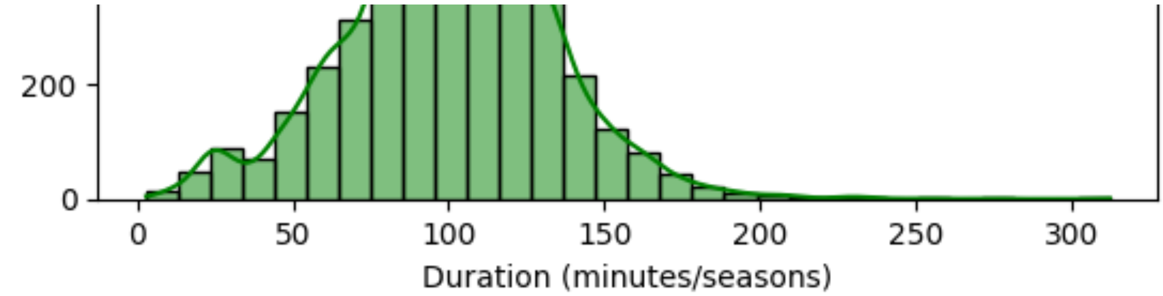
```
<ipython-input-22-142a6ffae63e>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs
  df_movies['numerical_duration1'] = df_movies['duration'].str.extract('(\d
<ipython-input-22-142a6ffae63e>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs
  df_tv_shows['numerical_duration1'] = df_tv_shows['duration'].str.extract(
```

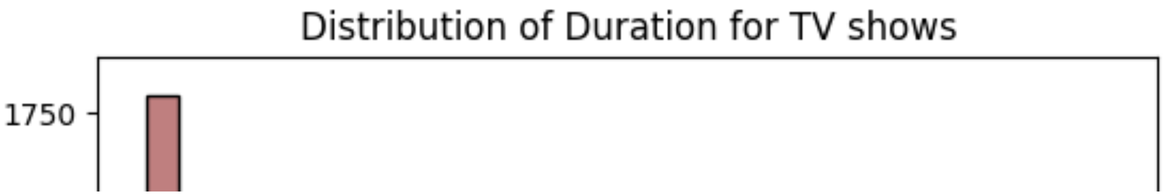
```
# Distplot for release year
sns.histplot(df['release_year'], kde=True, bins=30, color='gold')
plt.title("Distribution of Release Year")
plt.xlabel("Release Year")
plt.ylabel("Frequency")
plt.show()

# Distplot for duration for movies
sns.histplot(df_movies['numerical_duration1'], kde=True, color='green', bins=3(
plt.title("Distribution of Duration for Movies")
plt.xlabel("Duration (minutes/seasons)")
plt.ylabel("Frequency")
plt.show()
```

```
# Distplot for duration for movies
sns.histplot(df_tv_shows['numerical_duration1'], kde=True, color='maroon', bins
plt.title("Distribution of Duration for TV shows")
plt.xlabel("Duration (minutes/seasons)")
plt.ylabel("Frequency")
plt.show()
```










```
df_unnested = df.dropna(subset=['country']).copy()
df_unnested = df_unnested.assign(country=df_unnested['country'].str.split(', ')
df_unnested = df_unnested.explode('country')
df_unnested
```

	show_id	type	title	director	cast	country	date_added	rel
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	
7	s8	Movie	Sankofa	Haile Gerima	Kofi Ghanaba, Oyafunmike Ogunlano, Alexandra D...	United States	September 24, 2021	
7	s8	Movie	Sankofa	Haile Gerima	Kofi Ghanaba, Oyafunmike Ogunlano,	Ghana	September 24, 2021	

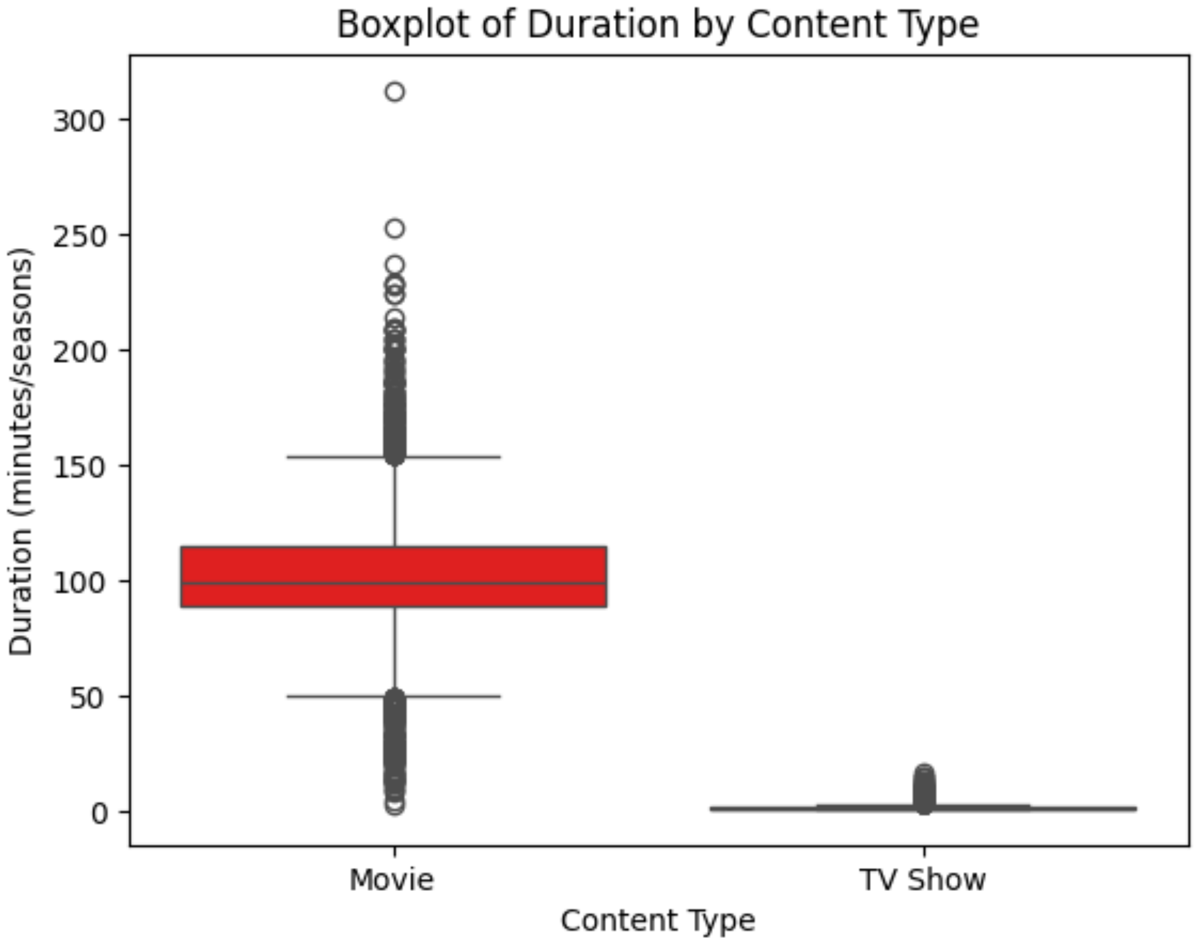
					Alexandra D...		
...	...	...	...	...	...	...	...
8801	s8802	Movie	Zinzana	Majid Al Ansari	Ali Suliman, Saleh Bakri, Yasa, Ali Al-Jabri, ...	Jordan	March 9, 2016
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	November 20, 2019
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	November 1, 2019
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019

10014 rows × 13 columns

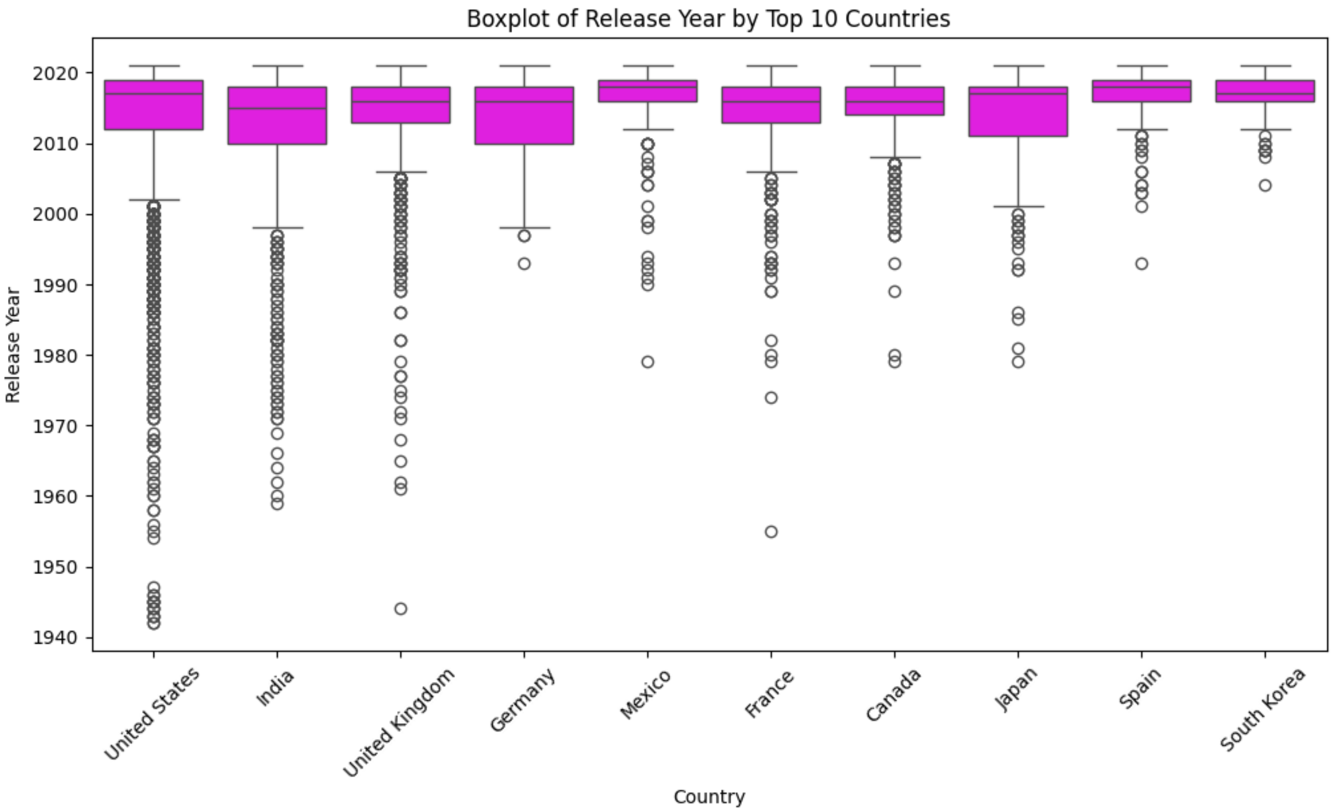
```
df_unnested['duration_numeric'] = df_unnested['duration'].str.extract('(\d+)').
df_unnested.head(5)
```

	show_id	type	title	director	cast	country	date_added	release_
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	
7	s8	Movie	Sankofa	Haile Gerima	Kofi Ghanaba, Oyafunmike Ogunlano, Alexandra D...	United States	September 24, 2021	
7	s8	Movie	Sankofa	Haile Gerima	Kofi Ghanaba, Oyafunmike Ogunlano, Alexandra D...	Ghana	September 24, 2021	

```
# Boxplot: Duration vs. Type
df_unnested = df_unnested.reset_index(drop=True)
sns.boxplot(data=df_unnested, x='type', y='duration_numeric', color='r')
plt.title("Boxplot of Duration by Content Type")
plt.xlabel("Content Type")
plt.ylabel("Duration (minutes/seasons)")
plt.show()
```



```
# Boxplot: Release year by Country (Top 10 Countries)
top_countries = df_unnested['country'].value_counts().head(10).index
filtered_data = df_unnested[df_unnested['country'].isin(top_countries)]
plt.figure(figsize=(12, 6))
sns.boxplot(data=filtered_data, x='country', y='release_year', color='magenta')
plt.title("Boxplot of Release Year by Top 10 Countries")
plt.xticks(rotation=45)
plt.xlabel("Country")
plt.ylabel("Release Year")
plt.show()
```



Step to handle the wrong ratings

Sol = Moved the rating to duration column using loc of pandas

```
df_unnested['rating'].unique()
```

```
array(['PG-13', 'TV-MA', 'TV-14', 'TV-Y7', 'PG', 'R', 'TV-PG', 'TV-Y',
       'TV-G', 'G', 'NC-17', '74 min', '84 min', '66 min', 'NR',
       'TV-Y7-FV', nan, 'UR'], dtype=object)
```

```
df[df['rating'] == '66 min']
```

	show_id	type	title	director	cast	country	date_added	release_yea
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	August 15, 2016	20

```
valid_ratings = [
    'PG-13', 'TV-MA', 'TV-14', 'TV-Y7', 'PG', 'R', 'TV-PG', 'TV-Y',
    'TV-G', 'G', 'NC-17', 'NR', 'TV-Y7-FV', 'UR'
]
```

```
invalid_rating_mask = ~df['rating'].isin(valid_ratings) & df['rating'].notna()
```

```
df.loc[invalid_rating_mask, 'duration'] = df.loc[invalid_rating_mask, 'rating']
```

```
df.loc[invalid_rating_mask, 'duration']
```

	duration
5541	74 min
5794	84 min
5813	66 min

dtype: object

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   show_id            8807 non-null   object
 1   type               8807 non-null   category
 2   title              8807 non-null   object
 3   director           6173 non-null   object
 4   cast               7982 non-null   object
 5   country            7976 non-null   category
 6   date_added         8797 non-null   object
 7   release_year       8807 non-null   int64
 8   rating             8803 non-null   object
 9   duration           8807 non-null   object
 10  listed_in          8807 non-null   object
 11  description        8807 non-null   object
 12  date_added_sorted  8709 non-null   datetime64[ns]
dtypes: category(2), datetime64[ns](1), int64(1), object(9)
memory usage: 804.9+ KB
```

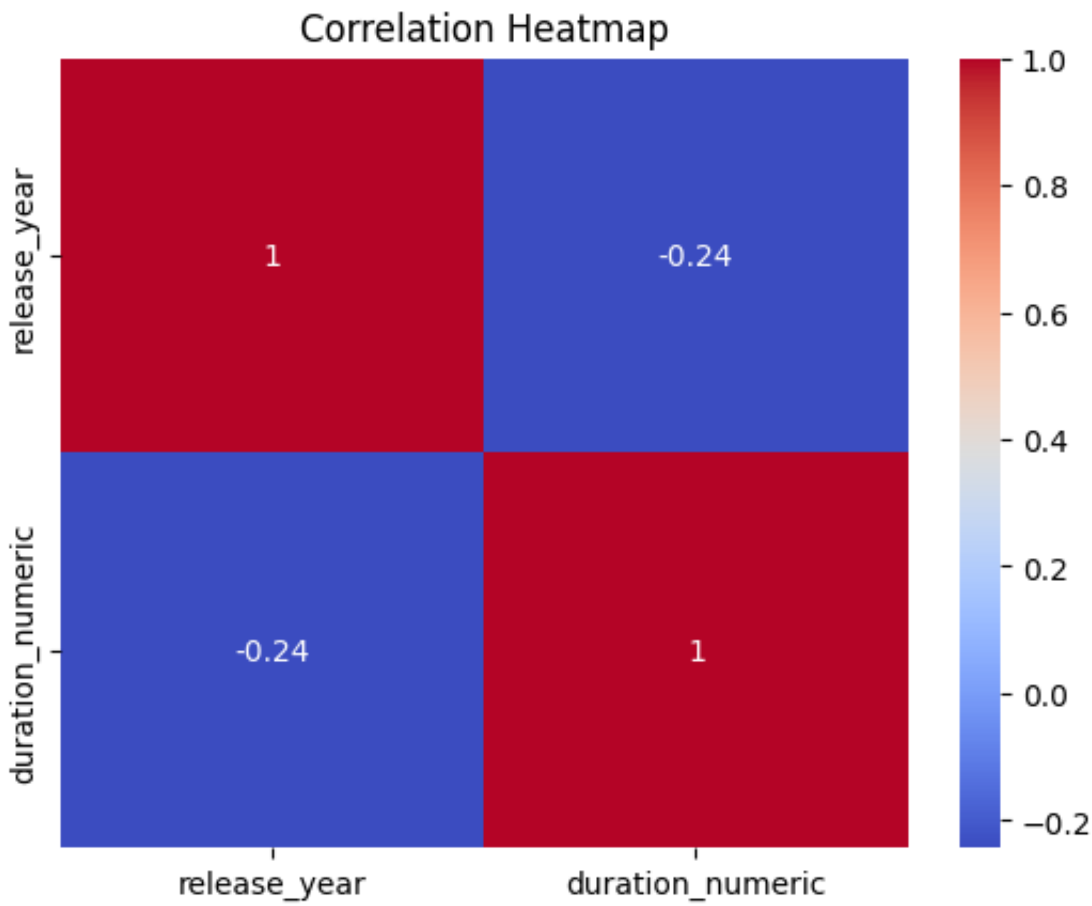
4.3 For Correlation: Heatmaps and Pairplots

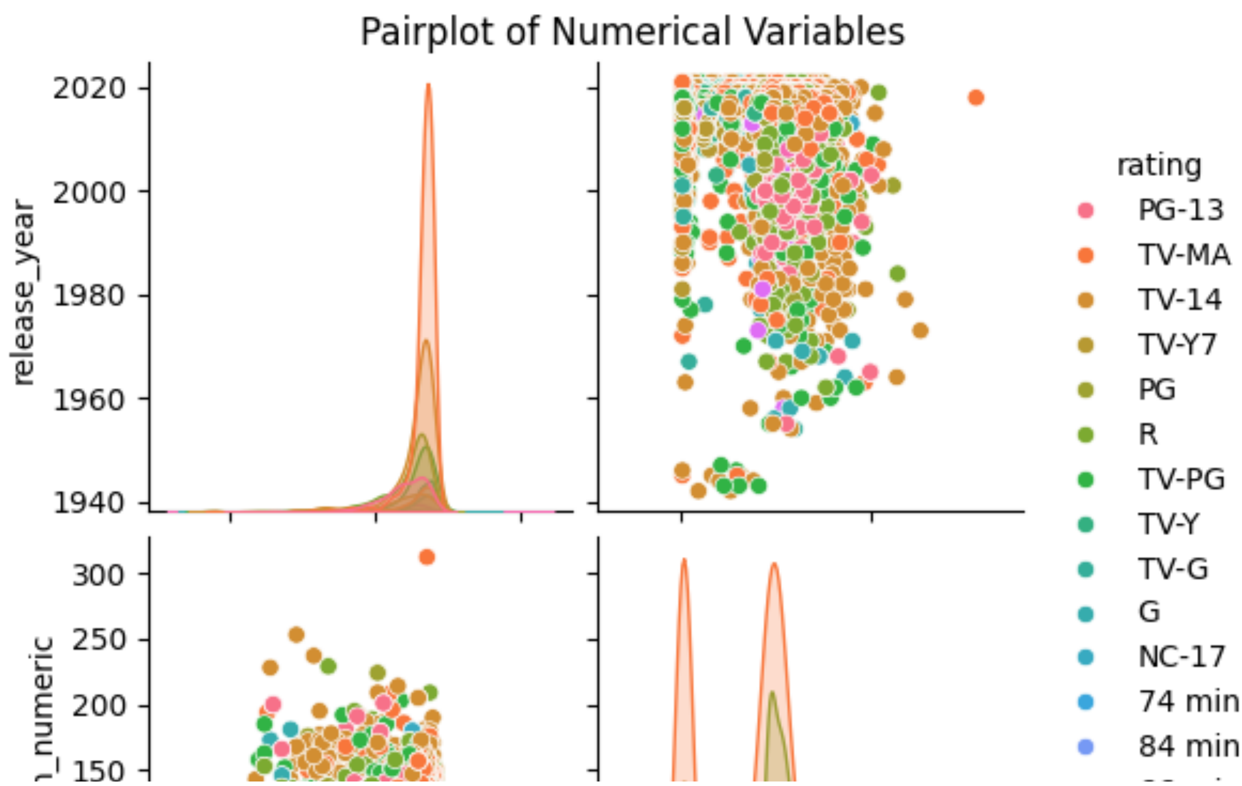
```
# Heatmap of correlation matrix
numerical_cols = ['release_year', 'duration_numeric']
correlation_matrix = df_unnested[numerical_cols].corr()
```

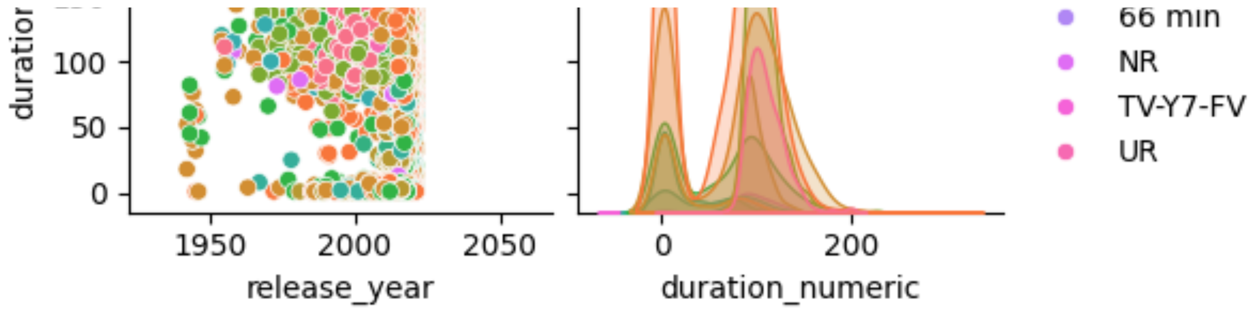
```
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
```

```
# Pairplot for numerical variables
```

```
sns.pairplot(df_unnested, hue='rating')
plt.suptitle("Pairplot of Numerical Variables", y=1.02)
plt.show()
```







5. Missing Value & Outlier check (Treatment optional) (10 Points)

Step 1: Missing Value Check

Missing values can impact data quality and analysis.

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

# Display missing value counts for each column
print("Missing Values per Column:\n")
print(missing_values)

# Percentage of missing values
missing_percentage = (missing_values / len(df)) * 100
print("\nPercentage of Missing Values:\n")
print(missing_percentage)
```

```
Missing Values per Column:

show_id                 0
type                    0
title                   0
director             2634
cast                  825
country               831
date_added             10
release_year            0
rating                  4
duration                0
listed_in               0
description             0
date_added_sorted      98
dtype: int64

Percentage of Missing Values:

show_id               0.000000
type                  0.000000
title                 0.000000
director             29.908028
cast                  9.367549
country               9.435676
date_added            0.113546
release_year          0.000000
rating                0.045418
duration              0.000000
listed_in             0.000000
description           0.000000
date_added_sorted     1.112751
dtype: float64
```

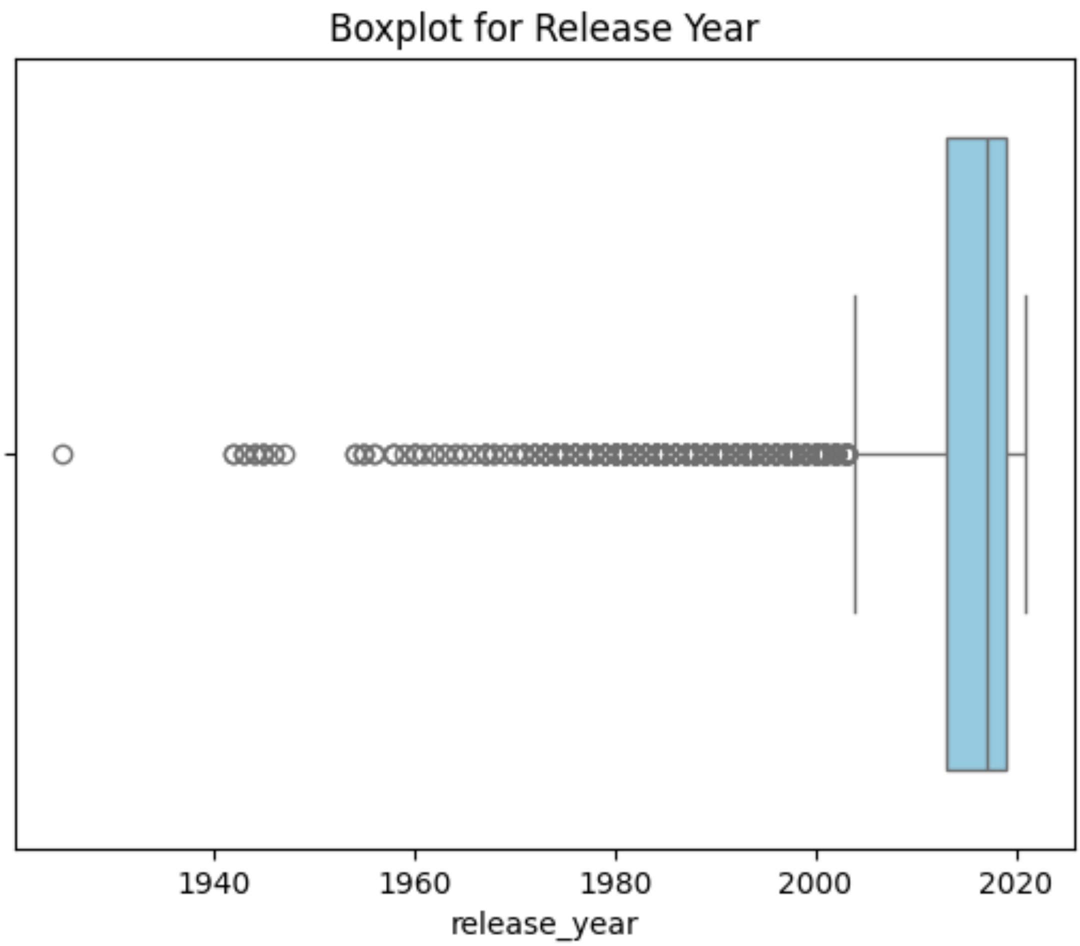
Insights -

If a column has too many missing values, consider dropping it or imputing values using mean, median, or mode. in this case there are no too many missing values for numerical features, so ignoring this step is fine.

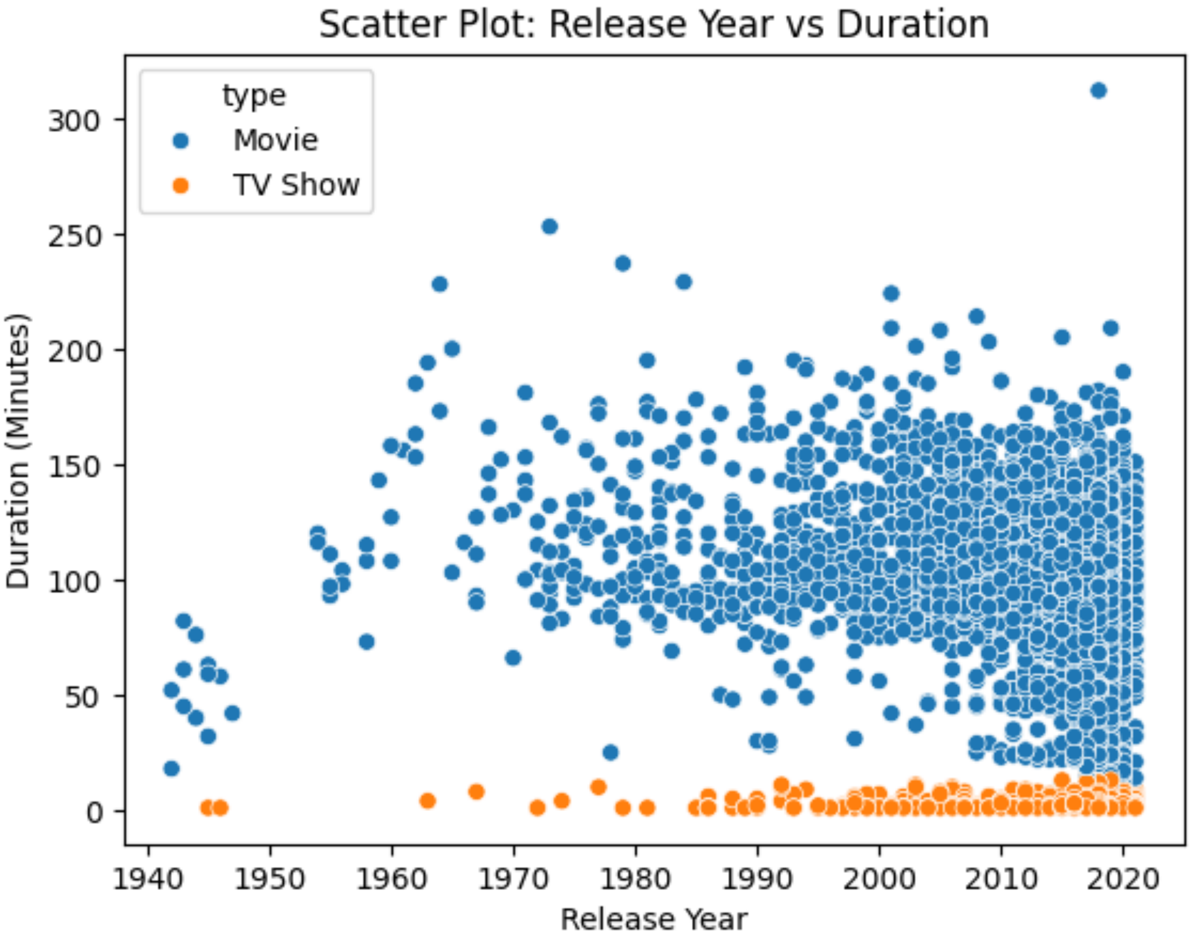
For categorical data, imputation could involve replacing missing values with the most frequent value. We can't replace 29% of the data with the most frequent value, that would result in completely wrong data.

Step 2: Outlier Detection

```
sns.boxplot(data=df, x='release_year', color='skyblue')
plt.title("Boxplot for Release Year")
plt.show()
```



```
# Scatter plot for detecting outliers in numerical data
sns.scatterplot(data=df_unnested, x='release_year', y='duration_numeric', hue='
plt.title("Scatter Plot: Release Year vs Duration")
plt.xlabel("Release Year")
plt.ylabel("Duration (Minutes)")
plt.show()
```



6. Insights based on Non-Graphical and Visual Analysis (10 Points)

Non-Graphical Analysis Insights

1. Content Type Distribution:

Movies dominate the dataset, comprising a significant majority compared to TV Shows.

Example: Movies: 70%, TV Shows: 30%.

2. Country Representation:

The United States is the most frequent producer of content, followed by India and the United Kingdom.

Large representation from Western countries compared to others.

3. Release Year Insights:

The mean release year is around 2014, with a significant number of releases after 2010.

The oldest content dates back to 1925.

4. Duration Patterns:

Movies have varied durations, while TV shows are listed in terms of seasons.

Outliers in movie durations and unusually long seasons for TV shows (e.g., 13+ seasons) are notable.

5. Missing Values:

The dataset has missing values, especially in columns like cast, director, and country, indicating incomplete metadata for some content.

Around 9.4% of the dataset lacks country information.

Visual Analysis Insights

Univariate Analysis

Distribution of Duration:

Movies: The duration follows a right-skewed distribution, with most movies lasting between 90–120 minutes.

Outliers exist for movies exceeding 3 hours.

Content Release Over Time:

A spike in content additions is observed post-2015, coinciding with Netflix's global expansion.

Bivariate Analysis

Type vs. Duration (Boxplot):

Movies generally have shorter durations (in minutes) compared to TV Shows (in seasons).

TV Shows with more than 5 seasons are relatively rare.

Country vs. Content Type (Grouped Bar Plot):

The US and India dominate movie production, whereas TV Shows have a more globally distributed production.

Multivariate Analysis

Correlation (Heatmap):

Release_year and Duration have no strong correlation, suggesting content duration isn't influenced by the release year.

Content type (type) significantly influences duration format.

```
Start coding or generate with AI.
```

7. Business Insights (10 Points) - Should include patterns observed in the data along with what you can infer from it

Content Creation Strategy: Focus on creating content in countries like India and the US, where Netflix has a large library and presumably high demand.

Customer Engagement: Target viewers with preferences for recent releases and short-duration content based on the observed trends.

Metadata Improvement: Address missing values, especially in critical fields like cast and director, country to enhance data-driven decision-making.

Outlier Analysis: Investigate unusually long durations or high season counts for potential data entry errors or special cases.

8. Recommendations (10 Points) - Actionable items for business. No technical jargon. No complications. Simple action items that everyone can understand

1. Invest More in Original Content from Popular Regions

Focus on creating more content in the United States, India, and the United Kingdom, as these countries dominate the existing library and likely have high viewer demand.

2. Expand TV Show Offerings Globally

Although movies dominate the platform, the demand for TV shows is steadily increasing. Expand the production of TV shows in emerging markets to attract new subscribers.

3. Focus on Recent Releases

Promote and prioritize content from recent years (2015 onwards), as users prefer newer shows and movies over older ones.

4. Balance Content Duration

Movies: Create content around the 90-120 minute mark, as most users prefer shorter, digestible movies. TV Shows: Focus on creating series with 3–5 seasons, as longer shows may deter new viewers.

5. Improve Metadata Completeness

Fill in missing data for directors, actors, and countries to make search and recommendations more accurate, enhancing the viewer experience.

6. Target Niche Genres

Increase production in popular genres like Drama, Comedy, and Action, while exploring niche genres to appeal to diverse audiences.

7. Optimize Global Launch Timing

Based on analysis, aim to release content in Q4 (September–December) to align with peak viewer engagement seasons (e.g., holidays).

8. Address Outliers

Investigate unusually long durations or high season counts in TV shows to determine if they are anomalies or unique features that can be marketed differently.

9. Localized Marketing Campaigns

Use insights about popular countries and genres to create regionalized campaigns tailored to viewer preferences in specific markets.

10. Continue Expanding the Global Library

As Netflix continues its global expansion, ensure a diverse library that represents local cultures and languages, attracting audiences from untapped regions.

End of Document