# Netflix Dataset Analysis - Complete Documentation

## Overview

This Jupyter Notebook performs comprehensive big-data analysis on a Netflix dataset using Python. The dataset contains information about TV Shows and Movies available on Netflix until 2021, sourced from Flixable (a third-party Netflix search engine) and available on Kaggle.

## Dataset Information

- **Source**: Flixable (third-party Netflix search engine)
- **Platform**: Kaggle
- **Coverage**: Netflix content up to 2021
- **Content Types**: TV Shows and Movies

## Required Libraries

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

## Initial Setup and Data Loading

### Loading the Dataset

```python
import pandas as pd
data = pd.read_csv(r"Netflix_Dataset.csv")
data
```

## Exploratory Data Analysis (EDA)

### Basic Dataset Information

#### 1. Head and Tail Operations

python

```python
# Display first 5 records
data.head()

# Display last 5 records
data.tail()
```

### 2. Dataset Shape and Size

python

```python
# Show number of rows and columns
data.shape

# Show total number of elements
data.size

# Show column names
data.columns

# Show data types of each column
data.dtypes

# Comprehensive dataset information
data.info()
```

## Data Cleaning Tasks

### Task 1: Duplicate Record Detection and Removal

**Objective**: Identify and remove duplicate records from the dataset.

python

```python
# Check for duplicate rows
data[data.duplicated()]

# Remove duplicates permanently
data.drop_duplicates(inplace=True)

# Verify removal
data[data.duplicated()]
data.shape
```

**Expected Output**: The shape will show reduced number of rows after duplicate removal.

### Task 2: Null Value Analysis with Heatmap

**Objective**: Identify null values and visualize them using a heatmap.

```python
# Check for null values
data.isnull()

# Count null values per column
data.isnull().sum()

# Visualize null values with heatmap
import seaborn as sns
sns.heatmap(data.isnull())
```

**Expected Output**: A heatmap showing null value distribution across columns, with darker areas indicating missing data.

## Analysis Questions and Solutions

### Q1: House of Cards Information

**Question**: For 'House of Cards', what is the Show ID and who is the Director?

```python
# Method 1: Using isin()
data[data['Title'].isin(['House of Cards'])]

# Method 2: Using str.contains()
data[data['Title'].str.contains('House of Cards')]
```

**Expected Output**: Complete record(s) for House of Cards showing Show ID, Director, and other details.

### Q2: Year with Highest Releases

**Question**: In which year were the highest number of TV Shows & Movies released? Show with Bar Graph.

python

```python
# Convert Release_Date to datetime
data['Date_N'] = pd.to_datetime(data['Release_Date'], errors='coerce')

# Count releases by year
data['Date_N'].dt.year.value_counts()

# Create bar graph
data['Date_N'].dt.year.value_counts().plot(kind='bar')
```

**Expected Output**:

- Value counts showing year-wise distribution
- Bar chart with years on x-axis and count on y-axis

## Q3: Movies vs TV Shows Distribution

**Question**: How many Movies & TV Shows are in the dataset? Show with Bar Graph.

python

```python
# Group by category and count
data.groupby('Category').Category.count()

# Create count plot
sns.countplot(data['Category'])
```

**Expected Output**:

- Count of Movies vs TV Shows
- Bar chart showing the distribution

## Q4: Movies Released in Year 2000

**Question**: Show all Movies released in year 2000.

python

```python
# Create Year column
data['Year'] = data['Date_N'].dt.year

# Filter for movies in 2000
data[(data['Category'] == 'Movie') & (data['Year'] == 2000)]

# Alternative: Movies in 2020
data[(data['Category'] == 'Movie') & (data['Year'] == 2020)]
```

**Expected Output**: Filtered dataset showing only movies from the specified year.

### Q5: Indian TV Shows

**Question**: Show only the Titles of all TV Shows released in India only.

```python
# Filter for Indian TV Shows and select only titles
data[(data['Category'] == 'TV Show') & (data['Country'] == 'India')]['Title']
```

**Expected Output**: List of TV show titles from India.

### Q6: Top 10 Directors

**Question**: Show Top 10 Directors who gave the highest number of TV Shows & Movies to Netflix.

```python
# Count content by director and show top 10
data['Director'].value_counts().head(10)
```

**Expected Output**: Ranked list of top 10 directors with their content count.

### Q7: Complex Filtering with AND/OR Operations

**Question**: Show all records where "Category is Movie and Type is Comedies" OR "Country is United Kingdom".

```python
# Filter for comedies OR UK content
data[(data['Category'] == 'Movie') & (data['Type'] == 'Comedies') | (data['Country'] == 'Unitec
```

**Expected Output**: Filtered dataset meeting the specified criteria.

### Q8: Tom Cruise Filmography

**Question**: In how many movies/shows was Tom Cruise cast?

python

```python
# Direct match (likely returns empty)
data[data['Cast'] == 'Tom Cruise']

# Search within cast strings
data[data['Cast'].str.contains('Tom Cruise', na=False)]

# Alternative approach with cleaned data
data_new = data.dropna()
data_new[data_new['Cast'].str.contains('Tom Cruise')]
```

**Expected Output**: Records containing Tom Cruise in the cast.

## Q9: Netflix Ratings Analysis

**Question**: What are the different ratings defined by Netflix?

python

```python
# Count unique ratings
data['Rating'].nunique()

# Show all unique ratings
data['Rating'].unique()
```

### Q9.1: TV-14 Movies in Canada

python

```python
# Count TV-14 movies in Canada
data[(data['Category'] == 'Movie') & (data['Rating'] == 'TV-14') & (data['Country'] == 'Canada'
```

### Q9.2: R-rated TV Shows after 2018

python

```python
# Find R-rated TV shows after 2018
data[(data['Category'] == 'TV Show') & (data['Rating'] == 'R') & (data['Year'] > 2018)]
```

**Expected Output**:

- List of all rating categories
- Specific counts for filtered criteria

## Q10: Maximum Duration Analysis

**Question**: What is the maximum duration of a Movie/Show on Netflix?

```python
# Check duration values
data.Duration.unique()

# Split duration into minutes and unit
data[['Minutes', 'Unit']] = data['Duration'].str.split(' ', expand=True)

# Convert to numeric and find statistics
data['Minutes'] = pd.to_numeric(data['Minutes'], errors='coerce')
data = data.dropna(subset=['Minutes'])

# Calculate statistics
data['Minutes'].max()
data['Minutes'].min()
data['Minutes'].mean()
```

**Expected Output**:

- Maximum, minimum, and average duration values
- Separate columns for duration number and unit

## Q11: Country with Most TV Shows

**Question**: Which individual country has the highest number of TV Shows?

```python
# Filter for TV shows only
data_tvshow = data[data['Category'] == 'TV Show']

# Count by country
data_tvshow.Country.value_counts()

# Show top country
data_tvshow.Country.value_counts().head(1)
```

**Expected Output**: Country with the highest TV show count.

## Q12: Sorting by Year

**Question**: How can we sort the dataset by Year?

python

```python
# Sort ascending
data.sort_values(by='Year')

# Sort descending
data.sort_values(by='Year', ascending=False).head(10)
```

**Expected Output**: Dataset sorted chronologically by release year.

### Q13: Complex Multi-Condition Filtering

**Question**: Find all instances where:

- Category is 'Movie' and Type is 'Dramas' OR
- Category is 'TV Show' & Type is 'Kids' TV'

python

```python
# Individual filters
data[(data['Category'] == 'Movie') & (data['Type'] == 'Dramas')].head(2)
data[(data['Category'] == 'TV Show') & (data['Type'] == "Kids' TV")]

# Combined filter
data[(data['Category'] == 'Movie') & (data['Type'] == 'Dramas') |
     (data['Category'] == 'TV Show') & (data['Type'] == "Kids' TV")]
```

**Expected Output**: Records matching either of the specified criteria.

## Key Pandas Operations Used

### Data Exploration

- `head()`, `tail()` - View data samples
- `shape`, `size` - Dataset dimensions
- `columns`, `dtypes`, `info()` - Metadata

### Data Cleaning

- `duplicated()`, `drop_duplicates()` - Handle duplicates
- `isnull()`, `dropna()` - Handle missing values
- `pd.to_datetime()`, `pd.to_numeric()` - Data conversion

### Data Analysis

- `value_counts()` - Count occurrences

- `groupby()` - Group operations
- `sort_values()` - Sort data
- `str.contains()`, `str.split()` - String operations
- Boolean indexing for filtering

### Visualization

- `plot(kind='bar')` - Bar charts
- `sns.heatmap()` - Heatmaps
- `sns.countplot()` - Count plots

## Best Practices Demonstrated

1. **Error Handling**: Using `errors='coerce'` in data conversion
2. **Missing Data**: Using `na=False` in string operations
3. **Data Integrity**: Checking for duplicates and null values
4. **Efficient Filtering**: Combining multiple conditions with & and |
5. **Data Visualization**: Using appropriate charts for different data types

## Conclusion

This analysis provides comprehensive insights into Netflix content distribution, including:

- Content volume trends over years
- Geographic distribution of content
- Director and cast analysis
- Content categorization and ratings
- Duration patterns

The notebook demonstrates essential data science skills including data cleaning, exploratory analysis, and visualization using pandas and seaborn libraries.