Netflix Dataset Analysis - Full Code and Line-by-Line Explanation

import pandas as pd

This line imports the pandas library using the alias 'pd'. Pandas is essential for working with tabular data such as CSV files. If this line is not included, you cannot load or manipulate datasets in a structured way. For example, you use 'pd.read\_csv()' to read a CSV file into a DataFrame.

import matplotlib.pyplot as plt

This imports the 'pyplot' module from the matplotlib library, using the alias 'plt'. It is used for creating static, animated, and interactive visualizations. If omitted, you can't generate graphs such as bar charts or histograms.

import seaborn as sns

Seaborn is a Python data visualization library based on matplotlib that provides a high-level interface for drawing attractive and informative graphics. It simplifies complex visualizations. If skipped, your plots might be less polished.

sns.set(style="whitegrid")

This sets the Seaborn theme to 'whitegrid', which helps make plots with a background grid, enhancing readability. If not set, default theme is used which might not be as readable.

plt.rcParams['figure.figsize'] = (12, 6)

Sets the default size of all plots to 12x6 inches. If not included, plots might be too small or inconsistent in size.

url = "https://raw.githubusercontent.com/hosna-tas/Netflix-Movies-and-TV-Shows-Project/main/data/netflix\_titles.csv"

Stores the URL of the dataset as a string variable. This makes it easy to load the dataset directly from the web without downloading it manually.

df = pd.read\_csv(url)

Reads the CSV file from the URL and loads it into a pandas DataFrame. This step is necessary to start analyzing the dataset.

df.head()

Displays the first 5 rows of the dataset, which gives a quick look at the structure and type of data.

df.isnull().sum()

Checks for missing values in each column. This is important for data cleaning. If skipped, you may encounter errors later during analysis.

df = df.dropna(subset=['director', 'country'])

Removes rows with missing 'director' or 'country'. This is important to ensure analysis based on these fields is accurate.

df['rating'].fillna('Unknown', inplace=True)

Fills missing ratings with 'Unknown' to prevent null values from affecting analysis. 'inplace=True' modifies the DataFrame directly.

df.drop\_duplicates(inplace=True)

Removes duplicate rows. Duplicate data can lead to incorrect analysis or biased results.

df['rating'].value\_counts()

Counts the occurrences of each unique rating value. Helps understand the distribution of ratings.

def map\_rating\_to\_age\_group(rating):

Defines a function to categorize ratings into age groups. This makes it easier to analyze consumer demographics.

df['age\_group'] = df['rating'].apply(map\_rating\_to\_age\_group)

Applies the function to each row in the 'rating' column and creates a new column 'age\_group'. This step is essential to analyze by age group.

df.head(10)

Shows the first 10 rows of the DataFrame, useful for verifying the new column was added correctly.

age\_group\_counts = df['age\_group'].value\_counts()

Counts how many times each age group appears in the dataset. This provides insight into the most common audience group.

sns.barplot(x=age\_group\_counts.index, y=age\_group\_counts.values)

Creates a bar plot of age group distribution using Seaborn. Helps visualize audience preference.

plt.title('Most Frequent Age Groups Watching Netflix')

Adds a title to the plot.

plt.xlabel('Age Group')

Labels the x-axis.

plt.ylabel('Number of Shows')

Labels the y-axis.

plt.show()

Displays the plot.

country\_age\_group\_counts = df.groupby(['country', 'age\_group']).size().reset\_index(name='count')

Groups the dataset by country and age group and counts the number of occurrences. This helps identify age trends across countries.

leading\_age\_group = country\_age\_group\_counts.loc[country\_age\_group\_counts.groupby('country')['count'].idxmax()]

Selects the most frequent age group per country.

top\_10 = leading\_age\_group.sort\_values(by='count', ascending=False).head(10)

Sorts and selects the top 10 countries by most dominant age group count.

sns.barplot(data=top\_10, y='country', x='count', hue='age\_group')

Plots the top 10 countries and their leading age groups.

plt.title('Top 10 Countries by Leading Age Group')

Adds a title to the country-wise age group plot.

plt.xlabel('Number of Shows')

Labels the x-axis for the above plot.

plt.ylabel('Country')

Labels the y-axis for the above plot.

plt.legend(title='Age Group')

Adds a legend to the plot.

plt.show()

Displays the second plot.

argentina\_df = df[df['country'] == 'Argentina']

Filters the dataset to only include data from Argentina for focused analysis.

top\_directors = argentina\_df['director'].value\_counts().head(3).reset\_index()

Finds the top 3 directors in Argentina.

top\_directors.columns = ['Director', 'Number of Shows']

Renames the columns for better readability.

print("Top 3 most preferred directors from Argentina:")

Outputs a message header.

print(top\_directors)

Displays the result of the top directors.

print("✅ Data analysis complete!")

Indicates successful completion.

print("📊 Total entries analyzed:", len(df))

Shows the total number of entries analyzed.

print("🌍 Countries covered:", df['country'].nunique())

Shows how many countries are included.

print("🎬 Total unique directors:", df['director'].nunique())

Displays the number of unique directors in the dataset.