Netflix Insights and Metrics

MILESTONE 1: Cleaning and Normalisation

Week 1: Cleaning

1. Dataset Overview

The dataset was loaded from `netflix_titles.csv` into a Pandas DataFrame. The initial shape of the dataset is (8807 rows, 12 columns). It includes 8,807 titles with 12 attributes:

- **title**
- **type** (Movie / TV Show)
- **director**
- **cast**
- **country**
- **date_added**
- **release_year**
- **rating**
- **duration**
- **listed_in (genres)**
- **description**

1.1. Data Cleaning Steps (Pandas)

1.11 Duplicate Removal

- Checked duplicates using df_read.duplicated().sum().
- Found **0 duplicates** and dropped them using:

df_read = df_read.drop_duplicates()

- The **shape of the dataset remained (8807, 12)**.

1.12 Missing Value Handling

- Identified missing values using *df_read.isnull().sum()*.
- Filled missing values in key fields with default indicators using the *.fillna()* function:

- Filled missing values in key fields:

    - `director` → *Unknown*
    - `cast` → *Not Available*
    - `country` → *Unknown*
    - `date_added` → *Not Available*
    - `rating` → *Not Rated*
    - `duration` → *Unknown*
- Dropped rows missing critical fields: `title`, `type`.

1.13 Standardization

- Trimmed whitespaces and normalized formatting:
    - `duration` cleaned (stripped, converted to Title Case).

      df_read['duration'] = df_read['duration'].str.strip().str.title()

    - `cast` standardized by stripping whitespace.

      df_read['cast'] = df_read['cast'].str.strip()

1.14 DataFrame Consolidation

- Saved the cleaned dataset as `df_clean`.
- The **final shape of the cleaned dataset (df_clean) is (8807 rows, 12 columns)**.
- Exported to CSV as **`netflix_cleaned.csv`** for downstream use with:

df_clean.to_csv("netflix_cleaned.csv", index=False)

1.2. Key Insights

All key insights were generated using the .value_counts() method in Pandas.

Content Distribution (Movies vs. TV Shows)

- Counted how many entries were **Movies** vs. **TV Shows**, showing Netflix's balance of formats using

type_distribution = df_clean['type'].value_counts()

- The analysis found: **6,131 Movies** (69.6%) and **2,676 TV Shows** (30.4%).

Top Directors

- Extracted the Top 10 directors using:

top_directors = df_clean['director'].value_counts().head(10)

- Top directors included **Rajiv Chilaka** (**19 titles**), **Raúl Campos, Jan Suter** (**18 titles**), and **Marcus Raboy** (**16 titles**).

Geographical Spread

- Extracted the Top 10 countries with most content using:

top_countries = df_clean['country'].value_counts().head(10)

- The top countries are the **United States** (**2,818 titles**), **India** (**972 titles**), and the **United Kingdom** (**419 titles**).

Ratings

- Listed the **Top 10 most common ratings**, highlighting Netflix's most frequent audience classifications using:

top_ratings = df_clean['rating'].value_counts().head(10)

- The most frequent audience classifications are **TV-MA** (**3,207 titles**), **TV-14** (**2,160 titles**), and **TV-PG** (**863 titles**).

1.3. Potential Applications

- **Content Strategy** → Use director and country-level insights to plan future acquisitions.
- **Genre & Rating Focus** → Explore dominant categories for personalized recommendations.
- **Regional Growth** → Understand high-content countries to strengthen global strategy.
- **Recommendation Systems** → Combine attributes like type, country, and rating to build content filters.

Week 2 Insights: Normalization

2. Data Normalization

To prepare the dataset for machine learning, categorical data was encoded into numerical formats, and key features were normalized. This process was performed on a copy of the cleaned DataFrame (df_clean).

2.1 Label Encoding for 'rating'

- Converted the categorical rating column into numerical labels using LabelEncoder.
- This assigns a unique integer to each rating category (e.g., TV-MA, TV-14), creating the new rating_label column.

label_encoder = LabelEncoder()

df_normalised['rating_label']=label_encoder.fit_transform(df_normalised['rating'].astype(str))

2.2 One-Hot Encoding for 'type'

- Applied OneHotEncoder to the type column to create binary features for 'Movie' and 'TV Show'.
- This avoids implying an incorrect ordinal relationship between the two categories.
- The resulting columns were merged back into the main DataFrame.

onehot_encoder = OneHotEncoder(sparse_output=False, drop=None)

type_encoded = onehot_encoder.fit_transform(df_normalised[['type']])

df_type_onehot = pd.DataFrame(type_encoded,

columns=onehot_encoder.get_feature_names_out(['type'])) df_normalised = pd.concat([df_normalised, df_type_onehot], axis=1)

2.3 Ordinal Encoding for 'country'

- Transformed the country column into ranked numerical values using OrdinalEncoder.

- The encoding order was based on the frequency of content from each country (value_counts()).
- A new country_ordinal column was added to the DataFrame.

country_order = df_normalised['country'].value_counts().index.tolist()

ordinal_encoder = OrdinalEncoder(categories=[country_order])

df_normalised['country_ordinal']=ordinal_encoder.fit_transform(df_normal ised[['country']])

2.4 Normalization (Min-Max Scaling)

- Applied MinMaxScaler to scale numerical features to a standard range between 0 and 1.
- This step is important for machine learning algorithms sensitive to the scale of input features.
- The following newly created columns were normalized:
    - **rating_label**
    - **country_ordinal**

scaler = MinMaxScaler() cols_to_normalize = ['rating_label', 'country_ordinal']

df_normalised[cols_to_normalize] = scaler.fit_transform(df_normalised[cols_to_normalize])

MILESTONE 2: EDA and Feature Engineering

Week 3: EDA

The Exploratory Data Analysis (EDA) was performed on the `netflix_cleaned.csv` DataFrame, which has a shape of **(8807 rows, 12 columns)**. The analysis focused on understanding content composition, temporal trends, and key descriptive statistics.

3.1. Data Quality Check

The cleaning process successfully handled most missing values, leaving only two columns with minor missing data that required further handling for temporal analysis:

- **Null Count:** `date_added` and the derived `year_added` each had **10 null values**.
- **Percentage of Nulls:** This represents a minimal **0.113546%** of the dataset.
- **Handling Code:** Missing values in the `year_added` column were filled with **0** for consistency:

df['year_added'] = df['year_added'].fillna(0)

3.2. Content Distribution Analysis

Content Type (Movies vs. TV Shows)

- **Insight:** Netflix's catalog is primarily composed of **Movies**, which account for over two-thirds of the content.
- **Distribution: Movies** make up **69.6%** (6,131 titles), while **TV Shows** make up **30.4%** (2,676 titles).

type_counts = df['type'].value_counts()
type_counts.plot(kind='pie', autopct='%1.1f%%', startangle=90)

Rating Distribution

- **Insight:** The most frequent audience classifications are those aimed at mature and young adult audiences.

- **Top Ratings:** The three most common ratings are **TV-MA** (Mature Audiences), **TV-14** (Teens 14+), and **TV-PG** (Parental Guidance Suggested).

rating_counts = df['rating'].value_counts()
rating_counts.plot(kind='bar')

3.3. Temporal & Categorical Insights

Content Growth Over Time

- **Insight:** The number of titles added to Netflix saw its steepest increase and peaked around **2019-2020**, indicating a period of aggressive content acquisition before a potential slowdown.

df['date_added'] = pd.to_datetime(df['date_added'], errors='coerce')
df['year_added'] = df['date_added'].dt.year
content_growth = df.groupby('year_added').size()

3.31. Top Genres

- **Insight:** When separating the comma-delimited listed_in column, **Dramas**, **Comedies**, and **International Movies/TV Shows** emerge as the dominant genres across the platform.

genres = df['listed_in'].dropna().str.split(',').explode().str.strip()
genre_counts = genres.value_counts().head(15)

3.32. Top Countries Contributing Content

- **Insight:** Content is heavily skewed towards the **United States**, which is the clear leader in contributions.

countries = df['country'].dropna().str.split(',').explode().str.strip()
country_counts = countries.value_counts().head(15)

3.4. Derived Feature Insights

3.41. Movie Length Categories

- The duration column (e.g., "90 Min") was cleaned and categorized into three bins: **Short** (0-60 min), **Medium** (60-120 min), and **Long** (120+ min).
- **Insight:** The majority of movies fall into the **Medium** length category (60-120 minutes).

```
movies['duration_num'] = (
    movies['duration'].str.extract('(\d+)').astype(float)
)
movies['length_category'] = pd.cut(
    movies['duration_num'],
    bins=[0, 60, 120, float('inf')],
    labels=['Short', 'Medium', 'Long']
)
```

3.42. Original vs. Licensed Content Proxy

- **Feature Engineering:** A binary feature, `is_original`, was created as a proxy for Netflix Original content by checking if the word "Netflix" (case-insensitive) was present in the title.
- **Insight:** This proxy feature shows the distribution of titles that explicitly contain "Netflix" in the title versus other licensed content.

```
df['is_original'] = df['title'].str.contains('(?i)netflix', na=False)
df['origin_label'] = df['is_original'].map({True: 'Original', False:
    'Licensed'})
```

BIVARIATE

3.5. Content Growth Over Time (Scatter Plot)

This plot shows the volume of new content (both Movies and TV Shows) added to Netflix each year, illustrating the platform's content acquisition strategy.

1. **Date Conversion:** The date_added column was converted to datetime objects, and the year was extracted into year_added.
2. **Grouping:** The data was **grouped by year_added** and the count of titles was calculated.
3. **Visualization:** A **scatter plot** was used to visualize the trend

```
df['date_added'] = pd.to_datetime(df['date_added'], errors='coerce')
df['year_added'] = df['date_added'].dt.year content_growth =
df.groupby('year_added').size().reset_index(name='count')
                                # ... plotting code ...
```

Key Insight

The chart clearly shows an **exponential growth** in content additions from around 2015, peaking sharply around **2019-2020**. This trend reflects Netflix's massive global expansion and investment in content during that period.

3.6. Movie Duration vs. Year Added (Seaborn Scatter Plot)

This plot filters for movies and explores the relationship between the year a movie was added and its running time, segmented by whether it's an "Original" or "Licensed" title.

1. **Filtering & Duration Extraction:** The DataFrame was filtered for 'Movie', and the numerical duration (minutes) was extracted into `duration_num`.
2. **Origin Label:** The `origin_label` column (Original/Licensed) was used for color coding. (This column was pre-calculated by checking for "Netflix" in the title).
3. **Visualization:** A **Seaborn scatter plot** compares year_added (X-axis) against duration_num (Y-axis), colored by `origin_label`.

```
movies = df[df['type'] == 'Movie'].copy()
movies['duration_num'] = (
    movies['duration'].str.extract('(\d+)').astype(float)
)
# ...
sns.scatterplot(
    data=movies,
    x='year_added',
    y='duration_num',
    hue='origin_label',
    alpha=0.7
)
```

Key Insight

The distribution of **Licensed** movies (yellow/green in the plot) covers a wider range of durations and release years, while **Original** movies (blue/red) are concentrated in **recent years (post-2015)** and primarily cluster around the **90 to 110-minute** mark. This suggests Netflix's internally branded movie production favors a standard, medium-length format.

3.7. Distribution of Movie Duration by Rating (Seaborn Histogram)

This stacked histogram visualizes the runtime distribution of movies and breaks down the total count by their MPAA/TV rating.

1. **Data:** Uses the movies DataFrame and the duration_num column.
2. **Visualization:** A **Seaborn stacked histogram** plots the movie duration distribution, with the bars segmented (stacked) by the rating category.

```
sns.histplot(
    data=movies,
    x='duration_num',
    hue='rating',
    multiple='stack',
    palette='Set1',
    bins=30
)
```

Key Insight

The majority of movies fall between **80 and 120 minutes**. Within this central peak, **TV-MA** and **TV-14** ratings dominate the volume, consistent with the overall high frequency of these ratings. This plot is essential for determining if extreme durations (very short or very long films) are predominantly associated with a specific audience rating (e.g., whether shorter content is mostly TV-Y)