Netflix Movie Data Analysis

This notebook provides a detailed analysis of Netflix movie data. We will clean the data, perform exploratory data analysis (EDA), and visualize key insights such as genre distribution, popularity, and release trends.

1. Import Libraries

We start by importing the necessary libraries for data manipulation and visualization.

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

2. Load Dataset

We load the dataset from `mymoviedb.csv`. The `lineterminator='\n'` argument is used to handle potential line break issues in the CSV file.

In [ ]:
```
df = pd.read_csv('mymoviedb.csv', lineterminator='\n')
df.head()
```

3. Data Exploration

Let's examine the structure of the dataset, including column names, data types, and non-null counts.

In [ ]:
```
df.info()
```

Initial Observations

- The dataset contains 9827 entries and 9 columns.
- Columns like `Release_Date` need to be converted to datetime objects.
- `Genre` contains comma-separated values which will need splitting.
- `Overview`, `Original_Language`, and `Poster_Url` might not be needed for this specific quantitative analysis.

In [ ]:
```
df['Genre'].head()
```

Check for duplicated rows to ensure data quality.

In [ ]:
```
df.duplicated().sum()
```

Get summary statistics for the numerical columns.

In [ ]:
```
df.describe()
```

4. Data Cleaning and Transformation

4.1 Date Conversion

We convert the `Release_Date` column to datetime objects and then extract just the year, as we are interested in annual trends.

In [ ]:
```
df['Release_Date'] = pd.to_datetime(df['Release_Date'])
print(df['Release_Date'].dtypes)
```

In [ ]:
```
df['Release_Date'] = df['Release_Date'].dt.year
df['Release_Date'].dtypes
```

4.2 Drop Unnecessary Columns

We drop columns that are not required for our analysis to keep the dataframe clean.

In [ ]:
```
cols = ['Overview', 'Original_Language', 'Poster_Url']
df.drop(cols, axis=1, inplace=True)
df.columns
```

4.3 Categorize Vote Average

We categorize the `Vote_Average` into bins: `not_popular`, `below_avg`, `average`, and `popular`. This helps in grouping movies based on audience reception.

In [ ]:
```
def categorize_col(df, col, labels):
    edges = [
        df[col].describe()['min'],
        df[col].describe()['25%'],
        df[col].describe()['50%'],
        df[col].describe()['75%'],
        df[col].describe()['max']
    ]
    df[col] = pd.cut(df[col], edges, labels=labels, duplicates='drop')
    return df
```

In [ ]:
```
labels = ['not_popular', 'below_avg', 'average', 'popular']
categorize_col(df, 'Vote_Average', labels)
df['Vote_Average'].unique()
```

Let's see the distribution of these new categories.

In [ ]:
```
df['Vote_Average'].value_counts()
```

4.4 Handle Missing Values

We drop any rows that still have missing values to ensure our analysis is robust.

In [ ]:
```
df.dropna(inplace=True)
df.isna().sum()
```

4.5 Explode Genre Column

The `Genre` column has multiple genres per movie (e.g., "Action, Adventure"). We split these strings and 'explode' the dataframe so that each row represents a single genre for a movie. This allows us to analyze genre frequency correctly.

In [ ]:
```
df['Genre'] = df['Genre'].str.split(', ')
df = df.explode('Genre').reset_index(drop=True)
df.head()
```

Convert `Genre` to a categorical type for better performance.

In [ ]:
```
df['Genre'] = df['Genre'].astype('category')
df.info()
```

5. Data Visualization & Analysis

Q1: What is the most frequent genre?

We visualize the count of movies for each genre.

In [ ]:
```
sns.set_style('whitegrid')
sns.catplot(y='Genre', data=df, kind='count',
            order=df['Genre'].value_counts().index,
            color='#4287F5')
plt.title('Genre Column Distribution')
plt.show()
```

Q2: Distribution of Vote Averages

We check how many movies fall into each of our defined vote categories.

In [ ]:
```
sns.catplot(y='Vote_Average', data=df, kind='count',
            order=df['Vote_Average'].value_counts().index,
            color='#4287F5')
plt.title('Vote Distribution')
plt.show()
```

Q3: Highest Popularity Movie

Let's find the movie with the maximum popularity score.

In [ ]:
```
df[df['Popularity'] == df['Popularity'].max()]
```

Q4: Lowest Popularity Movie

And the movie with the minimum popularity score.

In [ ]:
```
df[df['Popularity'] == df['Popularity'].min()]
```

Q5: Release Year Distribution

Which years had the most movie releases? We use a histogram to visualize this.

In [ ]:
```
df['Release_Date'].hist()
plt.title('Release Date Column Distribution')
plt.show()
```

Summary

- **Genre**: Drama is the most frequent genre.
- **Votes**: A significant portion of movies are rated as 'popular'.
- **Popularity**: We identified the specific movies with the highest and lowest popularity scores.
- **Release Trends**: We can observe the trend of movie releases over the years from the histogram.