

Netflix Movies/TV Shows — Exploratory Notebook (PYTHON)

This Jupyter notebook performs an exploratory analysis of the `netflix_movies_detailed_up_to_2025.csv` dataset. It inspects, cleans and visualizes metadata (title, type, release year, director, rating, popularity, revenue, country, genres, cast, etc.) and derives a few summary columns.

Libraries used

- pandas
- seaborn
- matplotlib.pyplot

(These are already imported in the notebook; do not re-import unless you intentionally want to overwrite definitions.)

Dataset

Filename: `netflix_movies_detailed_up_to_2025.csv`

Expected columns used by the notebook include: `title`, `type`, `date_added`, `release_year`, `rating`, `popularity`, `revenue`, `duration`, `country`, `genres`, `director`, `cast`, `language`, `budget`, `vote_count`, `vote_average` (not all columns are required to exist for all cells).

Notebook workflow (high level)

1. Load dataset:

- `df = pd.read_csv(r"netflix_movies_detailed_up_to_2025.csv")`

2. Quick inspection:

- `df`, `df.head()`, `df.tail()`, `df.shape`, `df.size`, `df.columns`, `df.dtypes`, `df.info()`

3. Duplicate handling:

- Detect duplicates: `df[df.duplicated()]`
- Remove duplicates permanently: `df.drop_duplicates(inplace=True)`

4. Missing values:

- Check and count: `df.isnull()` and `df.isnull().sum()`
- Visualize with heatmap: `sns.heatmap(df.isnull())`
- Drop column `duration` if it is fully null: `df_mod = df.drop('duration', axis=1)` (then re-check)

5. Searching and filtering:

- Exact title: `df[df['title'].isin(['X'])]`
- Title + year: `df[(df['title']=='X') & (df['release_year']==2022)]`
- Substring match: `df[df['title'].str.contains('X', na=False)]`
- Country/language/type filters, and complex boolean filters

6. Date handling:

- Convert `date_added` to datetime: `df['date_N'] = pd.to_datetime(df['date_added'])`
- Extract `Year`: `df['Year'] = df['date_N'].dt.year`
- Plot counts by year: `df['date_N'].dt.year.value_counts().plot(kind='bar')`

7. Numeric analyses and plots:

- Find rows with max values: `df.loc[df['rating'].idxmax()]`,
`df.loc[df['popularity'].idxmax()]`, `df.loc[df['revenue'].idxmax()]`
- Create readable revenue column: `df['Rev'] = (df['revenue'] / 1_000_000).apply(lambda x: f'{x:.3f}M')`
- Binning and plotting:
 - Ratings: `pd.cut(df['rating'], bins=[0,2,4,6,8,10])`
 - Popularity: `bins = list(range(0,101,10))` on `df['popularity']`
 - Revenue bins on `df['revenue'] / 1_000_000`

8. Grouping and counts:

- Count by `type`: `df.groupby('type').type.count()` and `sns.countplot(...)`
- Top directors: `df['director'].value_counts().head(15)` and threshold filters
- Country-wise counts for movies: `df_movie = df[df['type']=='Movie']` then `df_movie.country.value_counts()`

9. Sorting and more filters:

- Sort by `release_year`, multiple filters for genres, language, budget/votes thresholds

Notes & assumptions

- Some operations assume numeric types for `rating`, `popularity`, `revenue`. If these are strings, convert before numeric ops (e.g., `pd.to_numeric(..., errors='coerce')`).
- String operations use `na=False` to avoid errors when `cast` or `title` contains NaN.
- `drop_duplicates(inplace=True)` modifies `df` in place.
- The notebook creates new columns `date_N`, `Year`, and `Rev` that are reused by later cells.

How to run

1. Open the notebook in JupyterLab/Jupyter Notebook.
2. Make sure the CSV file is in the working directory or update the path.
3. Run cells from top to bottom (or use "Run All").
4. If plots do not display, ensure `%matplotlib inline` (or similar) is set in the notebook environment.

Common quick fixes

- If heatmap or seaborn plots look odd, verify there are no non-boolean/NaN issues in the `isnull()` data.

- If `pd.to_datetime()` fails, inspect `date_added` for inconsistent formats or missing values and use `errors='coerce'`.
- Convert columns before binning: `df['rating'] = pd.to_numeric(df['rating'], errors='coerce')`.