```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Reading CSV File
df = pd.read_csv("/content/netflix1.csv")
df.sample(5)
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 5,\n  \"fields\": [\n
{\n      \"column\": \"show_id\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 5,\n
\"samples\": [\n          \"s299\",\n          \"s7688\",\n
\"s1\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"type\",\n      \"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n          \"TV
Show\",\n          \"Movie\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"title\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 5,\n        \"samples\":
[\n          \"Quam's Money\",\n          \"P\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"director\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\": 5,\n
\"samples\": [\n          \"Kayode Kasum\",\n          \"Paul
Spurrier\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"country\",\n      \"properties\": {\n        \"dtype\": \"string\",\
n        \"num_unique_values\": 5,\n        \"samples\": [\n
\"Nigeria\",\n          \"United Kingdom\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"date_added\",\n
\"properties\": {\n        \"dtype\": \"object\",\n
\"num_unique_values\": 5,\n        \"samples\": [\n
\"8/6/2021\",\n          \"5/31/2019\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"release_year\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
5,\n        \"min\": 2006,\n        \"max\": 2020,\n
\"num_unique_values\": 4,\n        \"samples\": [\n          2020,\n
2006\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"rating\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n          \"TV-
14\",\n          \"TV-MA\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"duration\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 5,\n        \"samples\":
[\n          \"115 min\",\n          \"105 min\"\n        ],\n
```

```
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"listed_in\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 5,\n        \"samples\": [\n
\"Comedies, Dramas, International Movies\",\n          \"Horror
Movies, International Movies\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe"}
```


```
missing_values= df.isnull().sum()
print(missing_values)
```

```
show_id         0
type            0
title           0
director        0
country         0
date_added      0
release_year    0
rating          0
duration        0
listed_in       0
dtype: int64
```

Data is already clean

```
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8790,\n  \"fields\":
[\n    {\n      \"column\": \"show_id\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 8790,\n
\"samples\": [\n          \"s5505\",\n          \"s6264\",\n
\"s1463\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"type\",\n      \"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n          \"TV
Show\",\n          \"Movie\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"title\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 8787,\n
\"samples\": [\n          \"Your Excellency\",\n          \"Paradise
Lost\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"director\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 4528,\n
\"samples\": [\n          \"Ian Samuels\",\n          \"R\\u00e9my
Four, Julien War\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"country\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 86,\n
```

\"samples\": [\n   \"Guatemala\",\n   \"United States\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"date_added\",\n   \"properties\": {\n   \"dtype\": \"object\",\n   \"num_unique_values\": 1713,\n   \"samples\": [\n   \"12/21/2019\",\n   \"2/17/2017\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"release_year\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 8,\n   \"min\": 1925,\n   \"max\": 2021,\n   \"num_unique_values\": 74,\n   \"samples\": [\n   2013,\n   1977\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"rating\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 14,\n   \"samples\": [\n   \"G\",\n   \"NR\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"duration\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 220,\n   \"samples\": [\n   \"162 min\",\n   \"52 min\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"listed_in\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 513,\n   \"samples\": [\n   \"Romantic TV Shows, TV Comedies\",\n   \"Classic & Cult TV, TV Comedies\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
df.drop(["show_id"], axis=1, inplace=True)
```

```
df.shape
```

```
(8790, 9)
```

```
df.size
```

```
79110
```

```
df.index
```

```
RangeIndex(start=0, stop=8790, step=1)
```

```
df.columns
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8790 entries, 0 to 8789
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
```

```
 0   type          8790 non-null   object
 1   title         8790 non-null   object
 2   director      8790 non-null   object
 3   country       8790 non-null   object
 4   date_added    8790 non-null   object
 5   release_year  8790 non-null   int64
 6   rating        8790 non-null   object
 7   duration      8790 non-null   object
 8   listed_in     8790 non-null   object
dtypes: int64(1), object(8)
memory usage: 618.2+ KB
```

```
df["rating"].unique()
```

```
array(['PG-13', 'TV-MA', 'TV-PG', 'TV-14', 'TV-Y7', 'TV-Y', 'PG', 'TV-
G',
       'R', 'G', 'NC-17', 'NR', 'TV-Y7-FV', 'UR'], dtype=object)
```

converting date_added to date time format

```
df["date_added"] = df["date_added"].str.replace(",","")

df["date_added"] = pd.to_datetime(df["date_added"], format="mixed")

df["year"] = df["date_added"].dt.year

df["month"] = df["date_added"].dt.month_name()

df["date"] = df["date_added"].dt.day

df["genre"] = df["listed_in"].str.split(",").str[0]

df["genre"].value_counts().head(5)
```

```
genre
Dramas                    1599
Comedies                  1210
Action & Adventure         859
Documentaries              829
International TV Shows     773
Name: count, dtype: int64
```

```
movies_df = df[df["type"]=="Movie"]
movies_df.head(5)
```


```
{"summary":"{\n  \"name\": \"movies_df\",\n  \"rows\": 6126,\n
\"fields\": [\n    {\n      \"column\": \"type\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n
\"Movie\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

```
\"title\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 6124,\n        \"samples\": [\n
\"First Impression\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"director\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 4355,\n
\"samples\": [\n          \"Don Argott, Sheena M. Joyce\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"country\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 79,\n        \"samples\": [\n
\"Indonesia\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"date_added\",\n      \"properties\": {\n        \"dtype\":
\"date\",\n        \"min\": \"2008-01-01 00:00:00\",\n        \"max\":
\"2021-09-25 00:00:00\",\n        \"num_unique_values\": 1531,\n
\"samples\": [\n          \"2019-06-16 00:00:00\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"release_year\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
9,\n        \"min\": 1942,\n        \"max\": 2021,\n
\"num_unique_values\": 73,\n        \"samples\": [\n          2013\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"rating\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
14,\n        \"samples\": [\n          \"G\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"duration\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
205,\n        \"samples\": [\n          \"65 min\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"listed_in\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 278,\n        \"samples\": [\n
\"Documentaries, Sports Movies\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"year\",\n      \"properties\": {\n
\"dtype\": \"int32\",\n        \"num_unique_values\": 14,\n
\"samples\": [\n          2012\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"month\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 12,\n
\"samples\": [\n          \"March\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"date\",\n      \"properties\": {\n
\"dtype\": \"int32\",\n        \"num_unique_values\": 31,\n
\"samples\": [\n          28\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"genre\",\n      \"properties\": {\n        \"dtype\":
```

\"category\",\n   \"num_unique_values\": 19,\n
\"samples\": [\n   \"Documentaries\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"movies_df"}

```
movies_df = movies_df.copy()
movies_df.loc[:, 'duration_min'] = movies_df["duration"].str.split("
").str[0].astype(int)

movies_df.drop(["duration"], axis=1, inplace=True)
movies_df.head(5)
```

{"summary":"{\n \"name\": \"movies_df\",\n \"rows\": 6126,\n
\"fields\": [\n   {\n   \"column\": \"type\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 1,\n   \"samples\": [\n
\"Movie\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"title\",\n   \"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 6124,\n   \"samples\": [\n
\"First Impression\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"director\",\n   \"properties\": {\n   \"dtype\":
\"string\",\n   \"num_unique_values\": 4355,\n
\"samples\": [\n   \"Don Argott, Sheena M. Joyce\"\n   ],\
n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"country\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 79,\n   \"samples\": [\n
\"Indonesia\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"date_added\",\n   \"properties\": {\n   \"dtype\":
\"date\",\n   \"min\": \"2008-01-01 00:00:00\",\n   \"max\":
\"2021-09-25 00:00:00\",\n   \"num_unique_values\": 1531,\n
\"samples\": [\n   \"2019-06-16 00:00:00\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"release_year\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
9,\n   \"min\": 1942,\n   \"max\": 2021,\n
\"num_unique_values\": 73,\n   \"samples\": [\n   2013\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"rating\",\n   \"properties\":
{\n   \"dtype\": \"category\",\n   \"num_unique_values\":
14,\n   \"samples\": [\n   \"G\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"listed_in\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 278,\n   \"samples\": [\n
\"Documentaries, Sports Movies\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\

```
n    },\n    {\n      \"column\": \"year\",\n      \"properties\": {\n
\"dtype\": \"int32\",\n        \"num_unique_values\": 14,\n
\"samples\": [\n          2012\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"month\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 12,\n
\"samples\": [\n          \"March\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"date\",\n      \"properties\": {\n
\"dtype\": \"int32\",\n        \"num_unique_values\": 31,\n
\"samples\": [\n          28\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"genre\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 19,\n
\"samples\": [\n          \"Documentaries\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"duration_min\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
28,\n        \"min\": 3,\n        \"max\": 312,\n
\"num_unique_values\": 205,\n        \"samples\": [\n          65\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"movies_df"}
```


```
tv_shows_df = df[df["type"]=="TV Show"]
tv_shows_df.head(5)
```


```
{"summary":"{\n  \"name\": \"tv_shows_df\",\n  \"rows\": 2664,\n
\"fields\": [\n    {\n      \"column\": \"type\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          \"TV
Show\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"title\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 2663,\n        \"samples\": [\n
\"Tomorrow with You\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"director\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 226,\n
\"samples\": [\n          \"Luis Alfaro, Javier G\\u00f3mez
Santander\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"country\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 59,\n
\"samples\": [\n          \"France\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"date_added\",\n
\"properties\": {\n        \"dtype\": \"date\",\n        \"min\":
\"2008-02-04 00:00:00\",\n        \"max\": \"2021-09-24 00:00:00\",\n
\"num_unique_values\": 1012,\n        \"samples\": [\n
\"2019-11-12 00:00:00\"\n        ],\n        \"semantic_type\": \"\",\
```

n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"release_year\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 5,\n   \"min\": 1925,\n
\"max\": 2021,\n   \"num_unique_values\": 46,\n
\"samples\": [\n   1977\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"rating\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 9,\n   \"samples\":
[\n   \"NR\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"duration\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 15,\n
\"samples\": [\n   \"15 Seasons\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"listed_in\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 235,\n   \"samples\": [\n
\"International TV Shows\"\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"year\",\n   \"properties\": {\n   \"dtype\":
\"int32\",\n   \"num_unique_values\": 10,\n   \"samples\":
[\n   2014\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"month\",\n   \"properties\": {\n   \"dtype\": \"category\",\
n   \"num_unique_values\": 12,\n   \"samples\": [\n
\"November\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"date\",\n   \"properties\": {\n   \"dtype\": \"int32\",\n
\"num_unique_values\": 31,\n   \"samples\": [\n   5\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"genre\",\n   \"properties\":
{\n   \"dtype\": \"category\",\n   \"num_unique_values\":
17,\n   \"samples\": [\n   \"Crime TV Shows\"\n   ],\
n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   }\n]\n}","type":"dataframe","variable_name":"tv_shows_df"}

```
tv_shows_df = tv_shows_df.copy()
tv_shows_df.loc[:, "duration_season"] =
tv_shows_df["duration"].str.split(" ").str[0].astype(int)

tv_shows_df.drop(["duration"], axis=1, inplace=True)
tv_shows_df.head(5)
```

{"summary":"{\n \"name\": \"tv_shows_df\",\n \"rows\": 2664,\n
\"fields\": [\n   {\n   \"column\": \"type\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 1,\n   \"samples\": [\n   \"TV
Show\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"title\",\n   \"properties\": {\n   \"dtype\": \"string\",\n

```
\"num_unique_values\": 2663,\n        \"samples\": [\n
\"Tomorrow with You\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"director\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 226,\n
\"samples\": [\n          \"Luis Alfaro, Javier G\\u00f3mez
Santander\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"country\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 59,\n
\"samples\": [\n          \"France\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"date_added\",\n
\"properties\": {\n        \"dtype\": \"date\",\n        \"min\":
\"2008-02-04 00:00:00\",\n        \"max\": \"2021-09-24 00:00:00\",\n
\"num_unique_values\": 1012,\n        \"samples\": [\n
\"2019-11-12 00:00:00\"\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"release_year\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 5,\n        \"min\": 1925,\n
\"max\": 2021,\n        \"num_unique_values\": 46,\n
\"samples\": [\n          1977\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"rating\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 9,\n        \"samples\":
[\n          \"NR\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"listed_in\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 235,\n
\"samples\": [\n          \"International TV Shows\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"year\",\n      \"properties\": {\n
\"dtype\": \"int32\",\n        \"num_unique_values\": 10,\n
\"samples\": [\n          2014\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"month\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 12,\n
\"samples\": [\n          \"November\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"date\",\n      \"properties\": {\n
\"dtype\": \"int32\",\n        \"num_unique_values\": 31,\n
\"samples\": [\n          5\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"genre\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 17,\n
\"samples\": [\n          \"Crime TV Shows\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"duration_season\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
```

```
1,\n        \"min\": 1,\n        \"max\": 17,\n
\"num_unique_values\": 15,\n        \"samples\": [\n          15\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"tv_shows_df"}
```

```
df["type"].value_counts()
```

```
type
Movie      6126
TV Show    2664
Name: count, dtype: int64
```

Tells us how many movies are there as compared to TV shows.

```
df["rating"].value_counts()
```

```
rating
TV-MA       3205
TV-14       2157
TV-PG        861
R            799
PG-13        490
TV-Y7        333
TV-Y         306
PG           287
TV-G         220
NR            79
G             41
TV-Y7-FV       6
NC-17          3
UR             3
Name: count, dtype: int64
```

```
df["release_year"].value_counts().head(5)
```

```
release_year
2018    1146
2017    1030
2019    1030
2020     953
2016     901
Name: count, dtype: int64
```

```
df["country"].value_counts().head(5)
```

```
country
United States     3240
India             1057
United Kingdom     638
Pakistan           421
```

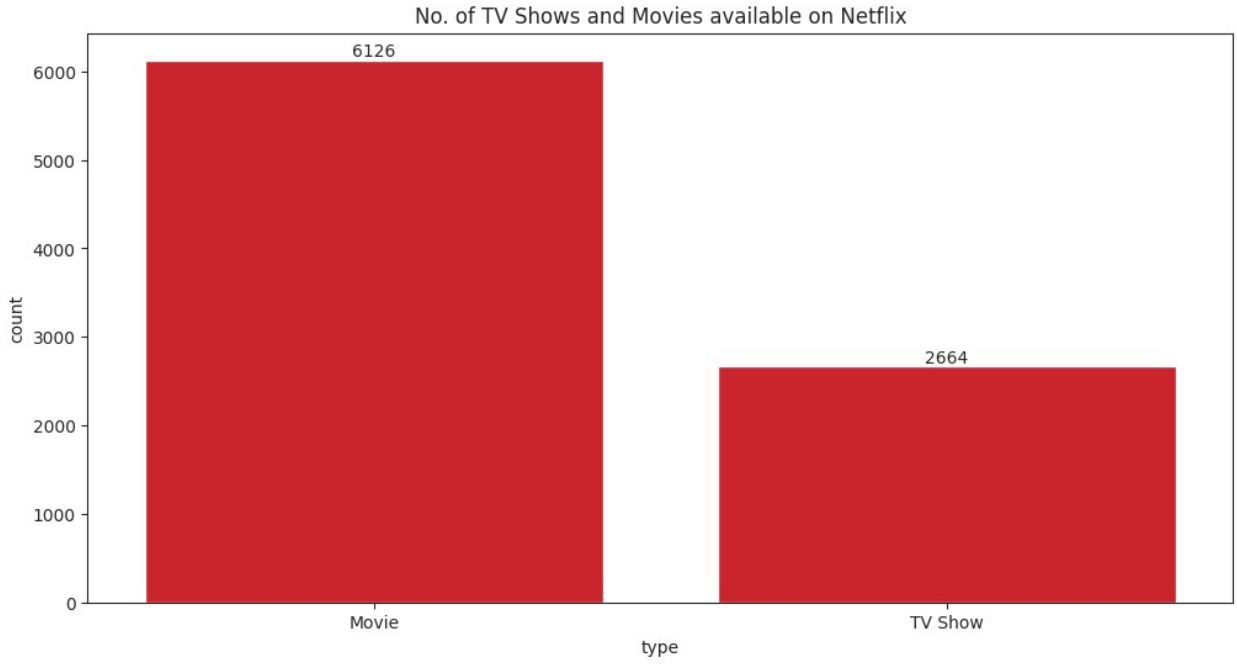
```
Not Given          287
Name: count, dtype: int64
```

```
df["genre"].value_counts().head(5)
```

```
genre
Dramas                     1599
Comedies                   1210
Action & Adventure          859
Documentaries               829
International TV Shows      773
Name: count, dtype: int64
```

```
sns.set_style("ticks")
```

```
fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.countplot(data=df, x="type", color="#E50914",
order=df["type"].value_counts().index)
ax.set_title("No. of TV Shows and Movies available on Netflix")
ax.bar_label(ax.containers[0], fontsize=10)
plt.show()
```

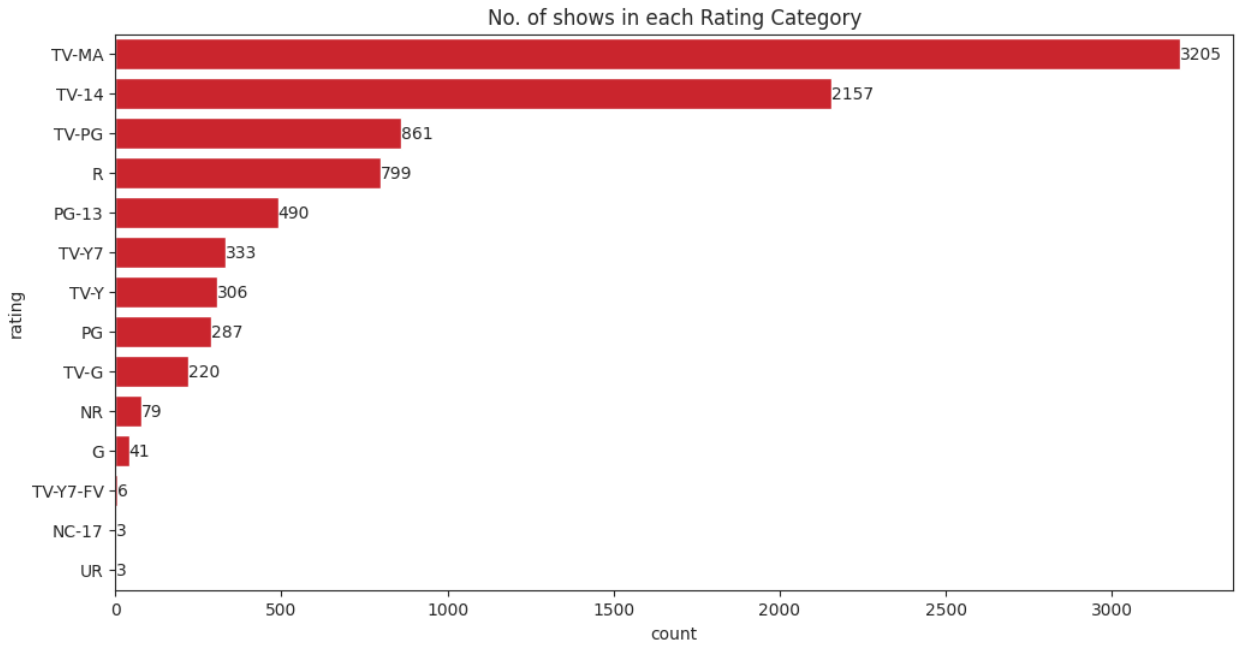


Insights

• Netflix has more number of Movies than TV Shows.

```
fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.countplot(data=df, y="rating",
order=df["rating"].value_counts().index, color="#E50914")
ax.set_title("No. of shows in each Rating Category")
```

```
ax.bar_label(ax.containers[0], fontsize=10)
plt.show()
```

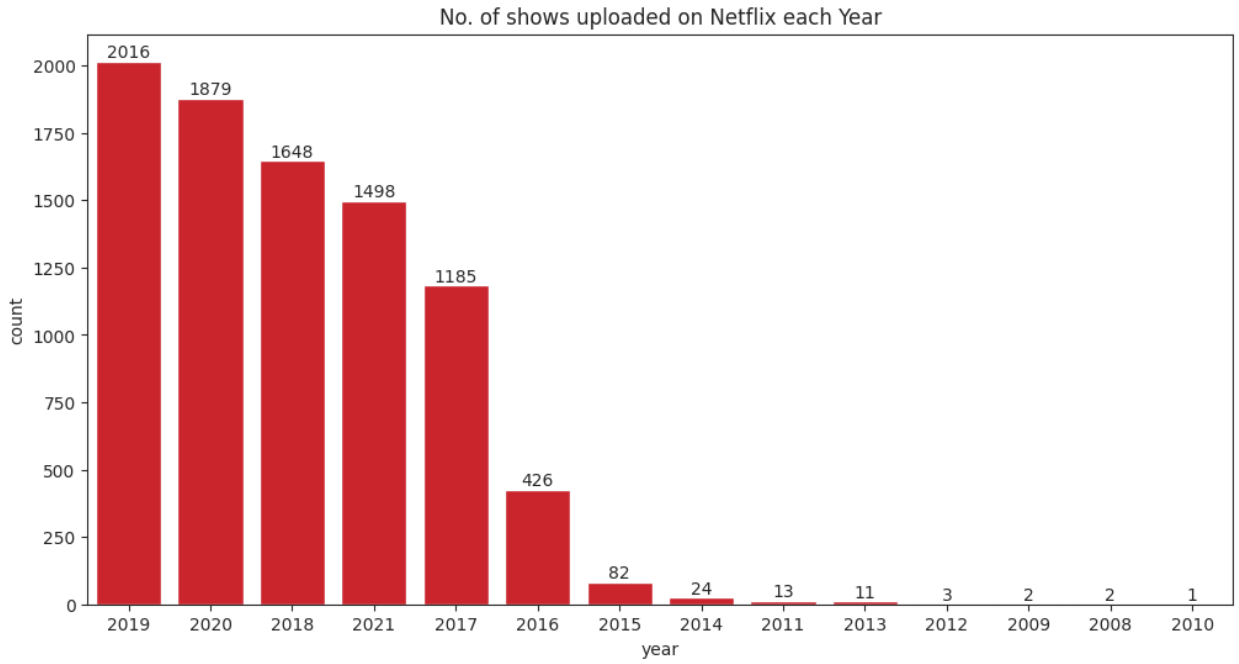


Insights

• Large number of shows on Netflix is rated under TV-MA. TV-MA : Mature Audience Only. Intended for adults and may be unsuitable for children under 17.

• Second largest collection of shows are rated under TV-14. TV-14 : Suitable for viewing by persons 14 years of age or older. Persons under 14 must be accompanied by an adult.

```
fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.countplot(data=df, x="year",
order=df["year"].value_counts().index, color="#E50914")
ax.set_title("No. of shows uploaded on Netflix each Year")
ax.bar_label(ax.containers[0], fontsize=10)
plt.show()
```



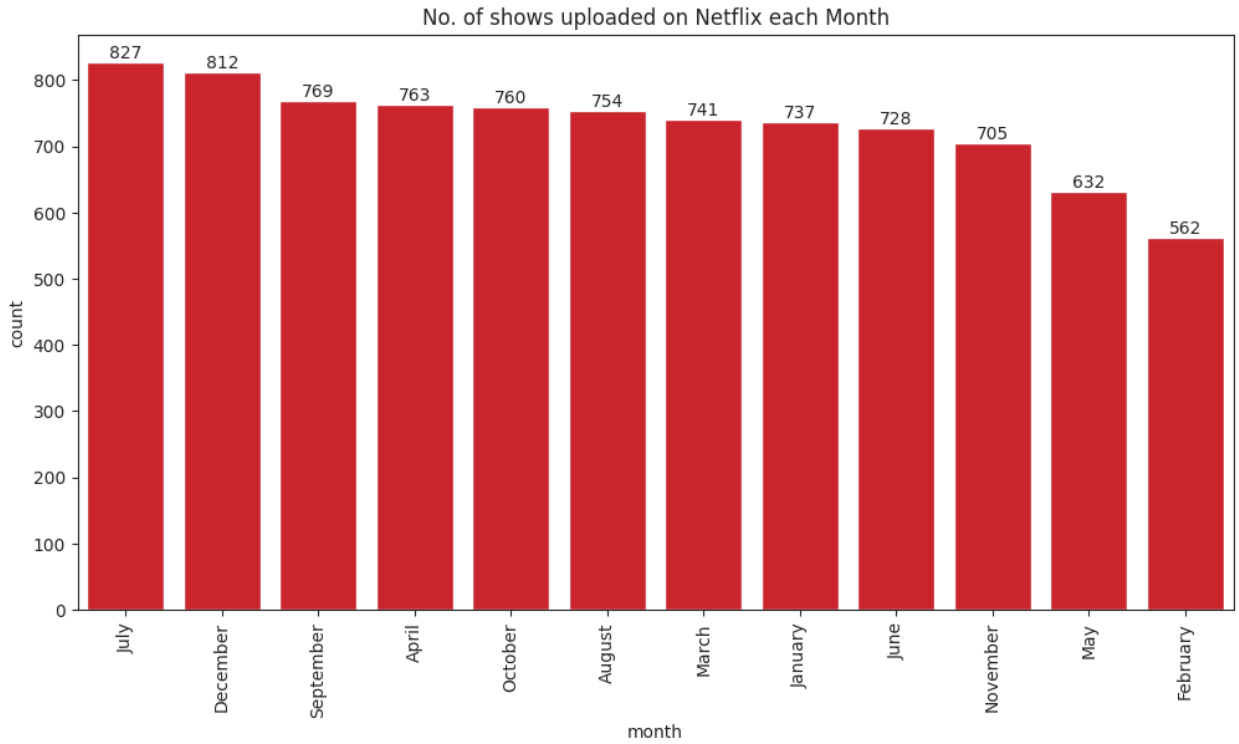
Insights

• Most number of shows on Netflix are uploaded in 2019.

• In years 2020, 2018, 2021, 2017, 2016, 2015 shows were uploaded heavily.

• Before 2015, Only few shows were uploaded.

```
fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.countplot(data=df, x="month",
order=df["month"].value_counts().index, color="#E50914")
ax.set_title("No. of shows uploaded on Netflix each Month")
ax.tick_params(axis='x', rotation=90)
ax.bar_label(ax.containers[0], fontsize=10)
plt.show()
```

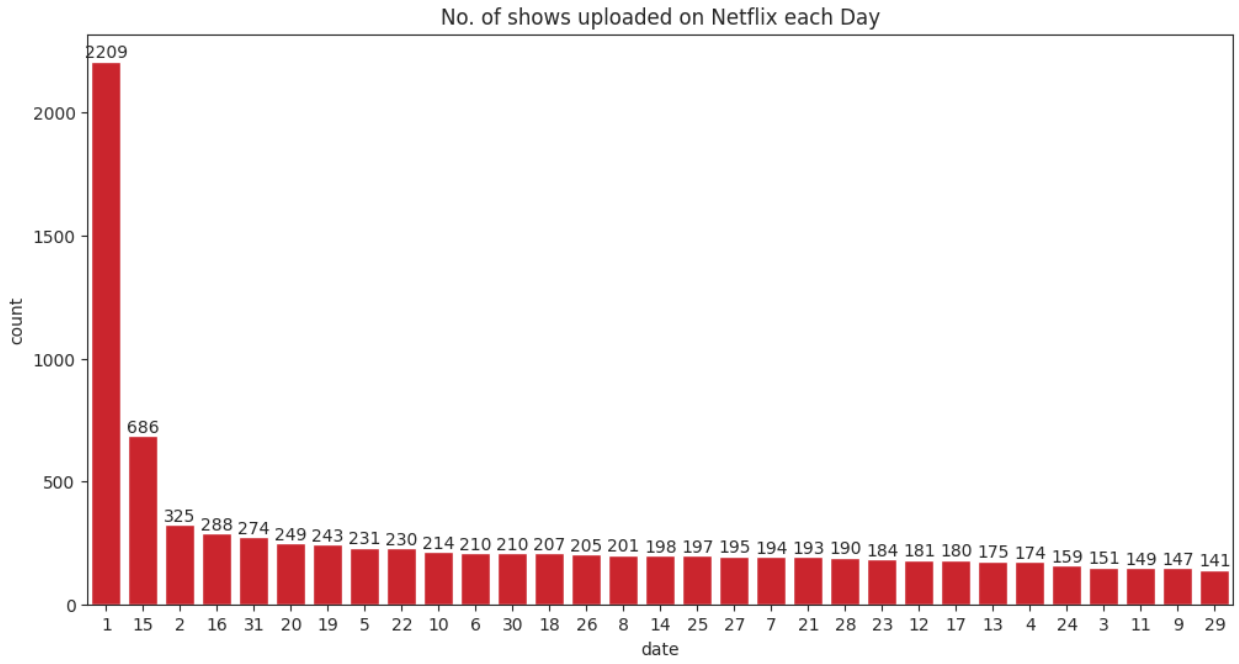


Insights

• In July and December month, most number of shows are uploaded on Netflix.

• Least shows are uploaded in February month.

```
fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.countplot(data=df, x="date",
order=df["date"].value_counts().index, color="#E50914")
ax.set_title("No. of shows uploaded on Netflix each Day")
ax.bar_label(ax.containers[0], fontsize=10)
plt.show()
```

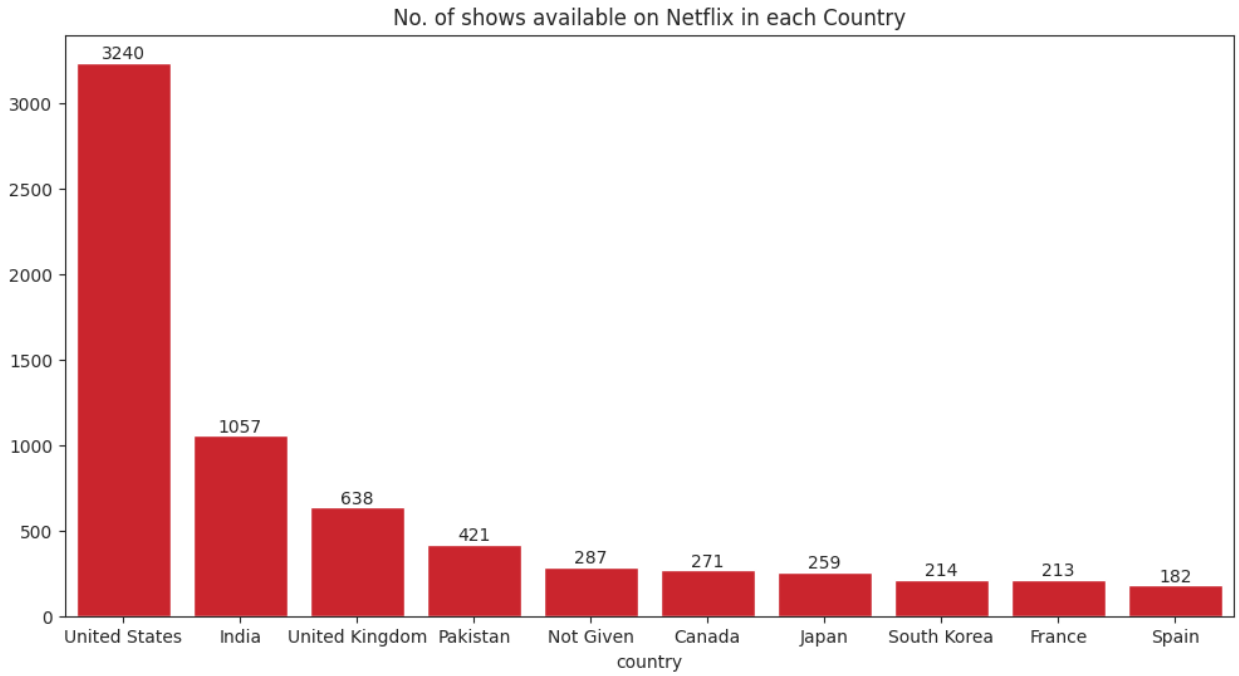


Insights

• In start and mid of the month, most number of shows are uploaded on Netflix.

• Least shows are uploaded near the end of the month.

```
country_counts = df["country"].value_counts().head(10)

fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.barplot(x=country_counts.index, y=country_counts.values,
color="#E50914")
ax.set_title("No. of shows available on Netflix in each Country")
ax.bar_label(ax.containers[0], fontsize=10)
plt.show()
```

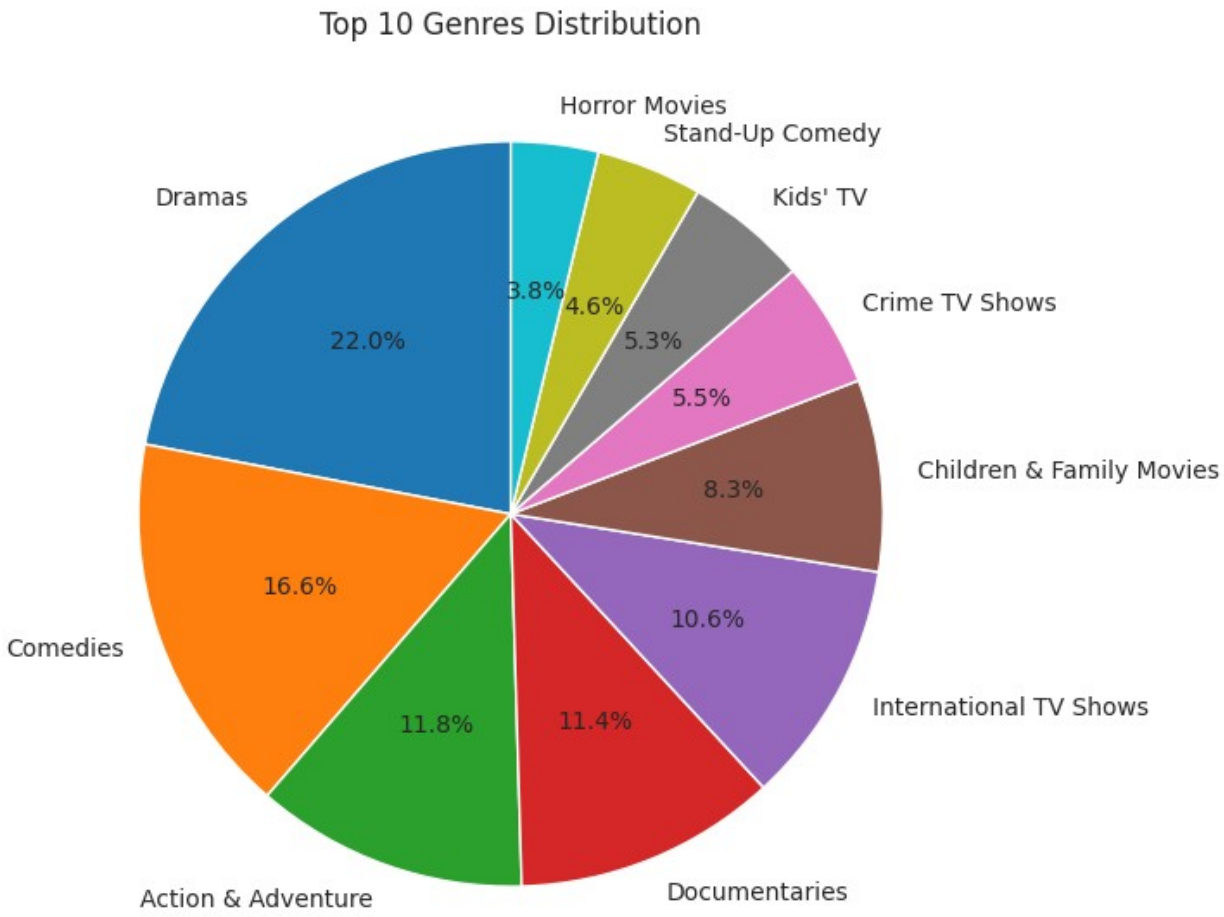


Insights

• Most number of shows are uploaded for United States.

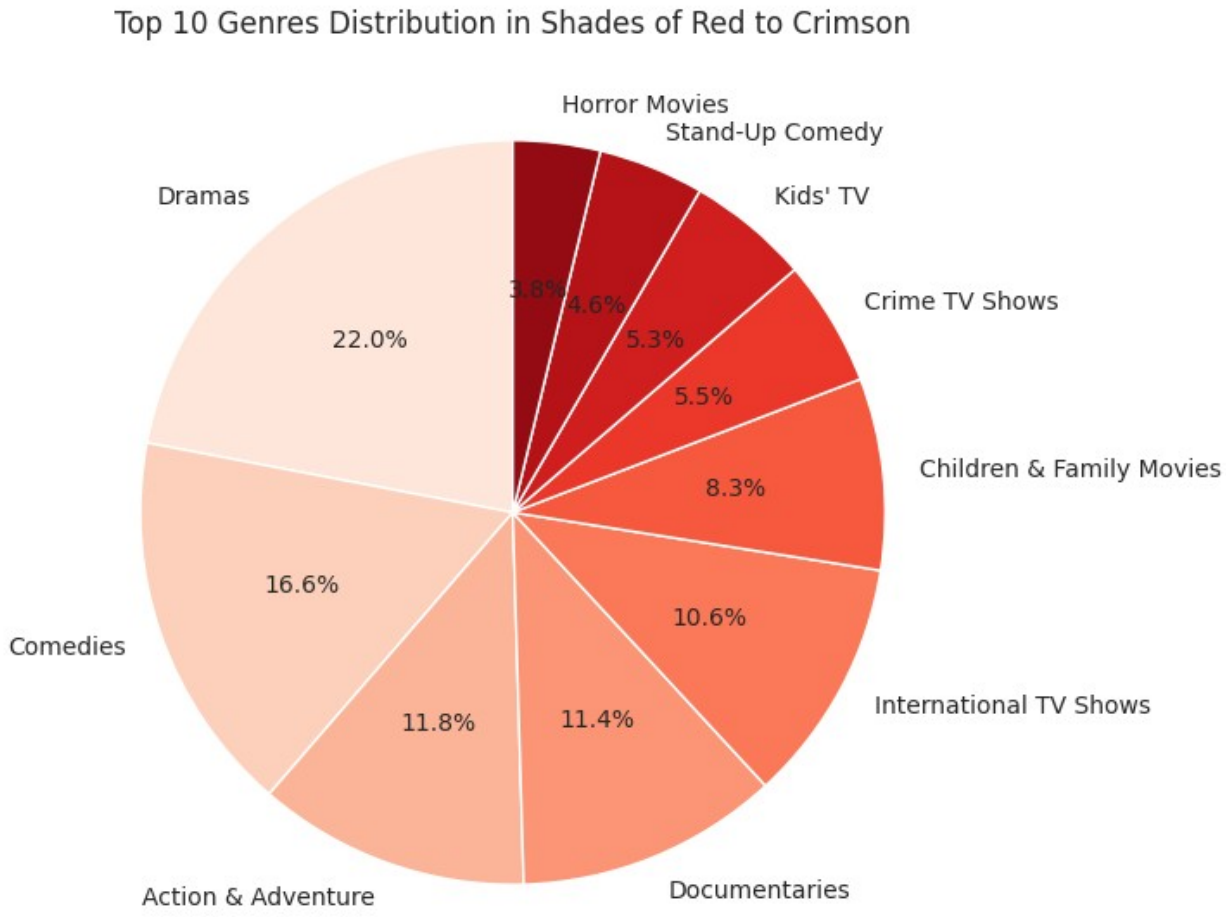
• India has the second largest collection of shows available on Netflix.

```
genre_counts = df['genre'].value_counts()

# Keep only the top 10 genres
top_10_genres = genre_counts.nlargest(10)

# Plotting the pie chart using matplotlib
plt.figure(figsize=(7, 7))  # Set figure size
plt.pie(top_10_genres, labels=top_10_genres.index, autopct='%1.1f%%',
startangle=90)

# Adding title
plt.title('Top 10 Genres Distribution')

# Show the plot
plt.show()
```




```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import matplotlib.colors as mcolors

# Count the occurrences of each genre
genre_counts = df['genre'].value_counts()

# Keep only the top 10 genres
top_10_genres = genre_counts.nlargest(10)

# Create a list of 10 different shades from red to crimson using
seaborn color_palette
colors = sns.color_palette("Reds", n_colors=10)

# Plotting the pie chart using matplotlib
plt.figure(figsize=(7, 7))  # Set figure size
plt.pie(top_10_genres, labels=top_10_genres.index, autopct='%1.1f%%',
startangle=90, colors=colors)
```

```
# Adding title
plt.title('Top 10 Genres Distribution in Shades of Red to Crimson')

# Show the plot
plt.show()
```

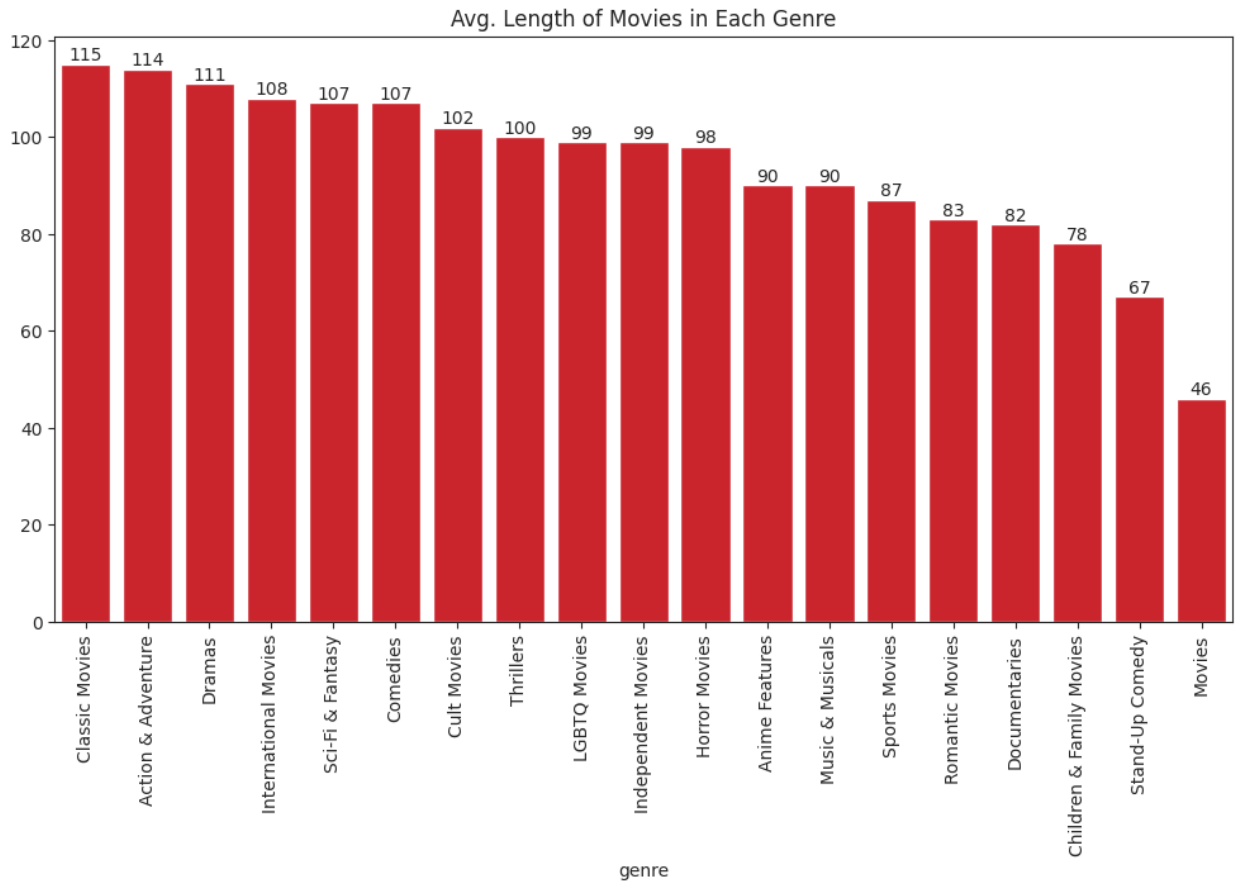


Insights

- Drama is the most popular genre
- Horror movies, although in the top 10 genres, are not watched a lot

```
genre_mean_movies = movies_df.groupby(["genre"])
["duration_min"].mean().round().sort_values(ascending=False)

fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.barplot(x=genre_mean_movies.index,
y=genre_mean_movies.values, color="#E50914")
ax.set_title("Avg. Length of Movies in Each Genre")
ax.tick_params(axis='x', rotation=90)
```

```
ax.bar_label(ax.containers[0], fontsize=10)
plt.show()
```

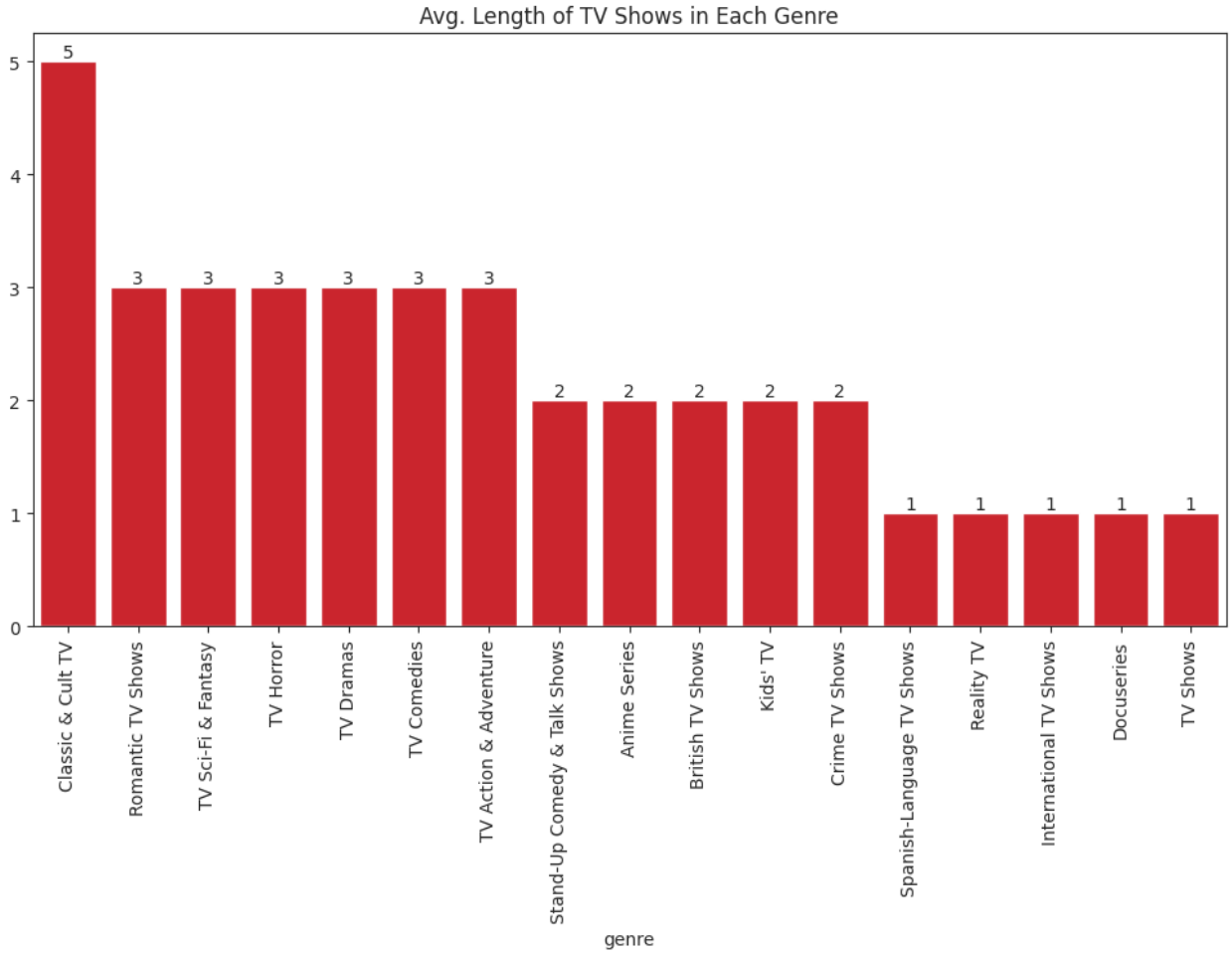


Insights

• Classic Movies on Netflix has the longest runtime approx (2 Hour 10 Minutes).

• Action and Adventure has the second longest runtime approx (2 Hours).

```
genre_mean_tv = tv_shows_df.groupby(["genre"])
["duration_season"].mean().round().sort_values(ascending=False)

fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.barplot(x=genre_mean_tv.index, y=genre_mean_tv.values,
color="#E50914")
ax.set_title("Avg. Length of TV Shows in Each Genre")
ax.tick_params(axis='x', rotation=90)
ax.bar_label(ax.containers[0], fontsize=10)
plt.show()
```



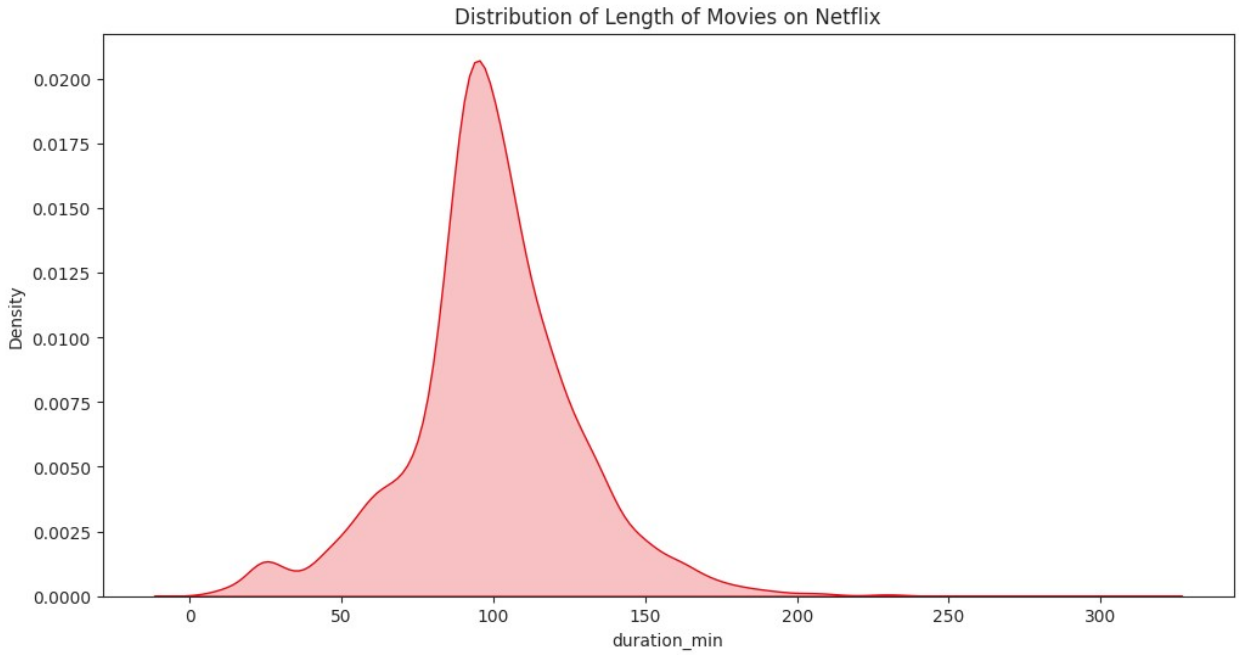
Avg. Length of TV Shows in each Genre

Insights

• Classic and Cult TV Shows on Netflix has the longest runtime approx (6 Seasons).

• Romantic, Fantasy, Horror, Drama, Comedy and Adventure has second largest runtime approx (3 Seasons).

```
fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.kdeplot(movies_df["duration_min"], color="#E50914",
fill=True)
ax.set_title("Distribution of Length of Movies on Netflix")
plt.show()
```

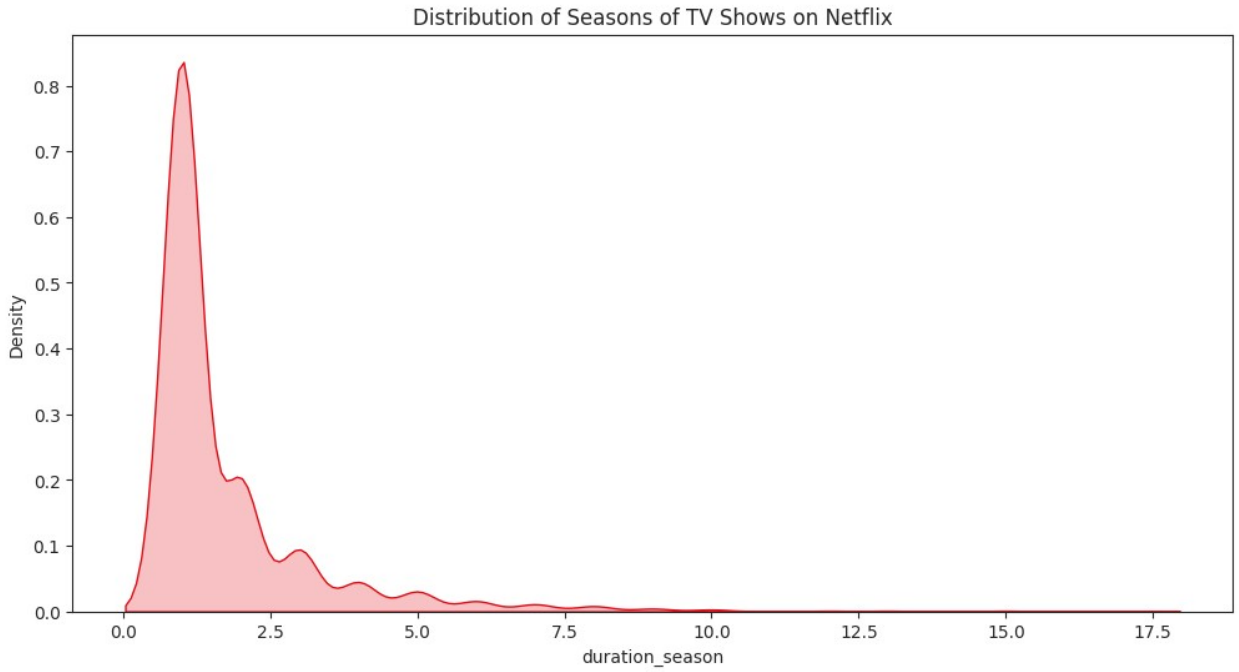


Insights

• Most number of Movies on Netflix has the runtime between 75 min to 125 min.

• Only few movies on Netflix are shorter than 75 min and larger than 125 min.

```
fig, ax = plt.subplots(figsize=(12, 6))
ax = sns.kdeplot(tv_shows_df["duration_season"], color="#E50914",
fill=True)
ax.set_title("Distribution of Seasons of TV Shows on Netflix")
plt.show()
```



Insights

• Most number of TV Shows on Netflix has the runtime between 1 to 2 Seasons.

• Few TV Shows on Netflix has runtime between 3 to 6 Seasons.

• Very few TV Shows has runtime more than 8 Seasons.

```
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8790,\n  \"fields\":
[\n    {\n      \"column\": \"type\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"TV Show\",\n          \"Movie\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"title\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 8787,\n        \"samples\": [\n          \"Your
Excellency\",\n          \"Paradise Lost\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"director\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\":
4528,\n        \"samples\": [\n          \"Ian Samuels\",\n
\"R\\u00e9my Four, Julien War\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"country\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
86,\n        \"samples\": [\n          \"Guatemala\",\n
\"United States\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

```
\"date_added\",\n      \"properties\": {\n        \"dtype\":
\"date\",\n        \"min\": \"2008-01-01 00:00:00\",\n        \"max\":
\"2021-09-25 00:00:00\",\n        \"num_unique_values\": 1713,\n
\"samples\": [\n          \"2019-12-21 00:00:00\",\n          \"2017-
02-17 00:00:00\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"release_year\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 8,\n        \"min\": 1925,\n
\"max\": 2021,\n        \"num_unique_values\": 74,\n
\"samples\": [\n          2013,\n          1977\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"rating\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
14,\n        \"samples\": [\n          \"G\",\n          \"NR\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"duration\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 220,\n        \"samples\": [\n          \"162
min\",\n          \"52 min\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"listed_in\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 513,\n
\"samples\": [\n          \"Romantic TV Shows, TV Comedies\",\n
\"Classic & Cult TV, TV Comedies\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"year\",\n      \"properties\": {\n
\"dtype\": \"int32\",\n        \"num_unique_values\": 14,\n
\"samples\": [\n          2012,\n          2009\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"month\",\n      \"properties\": {\
n        \"dtype\": \"category\",\n        \"num_unique_values\": 12,\
n        \"samples\": [\n          \"December\",\n
\"January\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"date\",\n      \"properties\": {\n        \"dtype\": \"int32\",\n
\"num_unique_values\": 31,\n        \"samples\": [\n          11,\n
14\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"genre\",\n      \"properties\": {\n        \"dtype\": \"category\",\
n        \"num_unique_values\": 36,\n        \"samples\": [\n
\"Sports Movies\",\n          \"TV Comedies\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```


```
<google.colab._quickchart_helpers.SectionTitle at 0x79e673d411e0>
```

```
from matplotlib import pyplot as plt
_df_0['release_year'].plot(kind='hist', bins=20, title='release_year')
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
from matplotlib import pyplot as plt
_df_1['date'].plot(kind='hist', bins=20, title='date')
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x79e6761bc7f0>
```

```
from matplotlib import pyplot as plt
import seaborn as sns
_df_2.groupby('type').size().plot(kind='barh',
color=sns.palettes.mpl_palette('Dark2'))
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
from matplotlib import pyplot as plt
import seaborn as sns
_df_3.groupby('title').size().plot(kind='barh',
color=sns.palettes.mpl_palette('Dark2'))
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
from matplotlib import pyplot as plt
import seaborn as sns
_df_4.groupby('director').size().plot(kind='barh',
color=sns.palettes.mpl_palette('Dark2'))
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
from matplotlib import pyplot as plt
import seaborn as sns
_df_5.groupby('country').size().plot(kind='barh',
color=sns.palettes.mpl_palette('Dark2'))
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x79e670633130>
```

```
from matplotlib import pyplot as plt
_df_6.plot(kind='scatter', x='release_year', y='date', s=32, alpha=.8)
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x79e670696dd0>
```

```
from matplotlib import pyplot as plt
import seaborn as sns
def _plot_series(series, series_name, series_index=0):
  palette = list(sns.palettes.mpl_palette('Dark2'))
  xs = series['date_added']
  ys = series['release_year']

  plt.plot(xs, ys, label=series_name, color=palette[series_index %
len(palette)])

fig, ax = plt.subplots(figsize=(10, 5.2), layout='constrained')
df_sorted = _df_7.sort_values('date_added', ascending=True)
for i, (series_name, series) in enumerate(df_sorted.groupby('type')):
```

```
  _plot_series(series, series_name, i)
  fig.legend(title='type', bbox_to_anchor=(1, 1), loc='upper left')
sns.despine(fig=fig, ax=ax)
plt.xlabel('date_added')
_ = plt.ylabel('release_year')
```

```
from matplotlib import pyplot as plt
import seaborn as sns
def _plot_series(series, series_name, series_index=0):
  palette = list(sns.palettes.mpl_palette('Dark2'))
  xs = series['date_added']
  ys = series['release_year']

  plt.plot(xs, ys, label=series_name, color=palette[series_index %
len(palette)])

fig, ax = plt.subplots(figsize=(10, 5.2), layout='constrained')
df_sorted = _df_8.sort_values('date_added', ascending=True)
for i, (series_name, series) in enumerate(df_sorted.groupby('title')):
  _plot_series(series, series_name, i)
  fig.legend(title='title', bbox_to_anchor=(1, 1), loc='upper left')
sns.despine(fig=fig, ax=ax)
plt.xlabel('date_added')
_ = plt.ylabel('release_year')
```

```
from matplotlib import pyplot as plt
import seaborn as sns
def _plot_series(series, series_name, series_index=0):
  palette = list(sns.palettes.mpl_palette('Dark2'))
  xs = series['date_added']
  ys = series['release_year']

  plt.plot(xs, ys, label=series_name, color=palette[series_index %
len(palette)])

fig, ax = plt.subplots(figsize=(10, 5.2), layout='constrained')
df_sorted = _df_9.sort_values('date_added', ascending=True)
for i, (series_name, series) in
enumerate(df_sorted.groupby('director')):
  _plot_series(series, series_name, i)
  fig.legend(title='director', bbox_to_anchor=(1, 1), loc='upper
left')
sns.despine(fig=fig, ax=ax)
plt.xlabel('date_added')
_ = plt.ylabel('release_year')
```

```
from matplotlib import pyplot as plt
import seaborn as sns
def _plot_series(series, series_name, series_index=0):
  palette = list(sns.palettes.mpl_palette('Dark2'))
```

```
  xs = series['date_added']
  ys = series['release_year']

  plt.plot(xs, ys, label=series_name, color=palette[series_index %
len(palette)])

fig, ax = plt.subplots(figsize=(10, 5.2), layout='constrained')
df_sorted = _df_10.sort_values('date_added', ascending=True)
for i, (series_name, series) in
enumerate(df_sorted.groupby('country')):
  _plot_series(series, series_name, i)
  fig.legend(title='country', bbox_to_anchor=(1, 1), loc='upper left')
sns.despine(fig=fig, ax=ax)
plt.xlabel('date_added')
_ = plt.ylabel('release_year')
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x79e670697970>
```

```
from matplotlib import pyplot as plt
_df_11['release_year'].plot(kind='line', figsize=(8, 4),
title='release_year')
plt.gca().spines[['top', 'right']].set_visible(False)
```

```
from matplotlib import pyplot as plt
_df_12['date'].plot(kind='line', figsize=(8, 4), title='date')
plt.gca().spines[['top', 'right']].set_visible(False)
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x79e673d417b0>
```

```
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['title'].value_counts()
    for x_label, grp in _df_13.groupby('type')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('type')
_ = plt.ylabel('title')
```

```
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['director'].value_counts()
    for x_label, grp in _df_14.groupby('title')
})
sns.heatmap(df_2dhist, cmap='viridis')
```

```
plt.xlabel('title')
_ = plt.ylabel('director')
```

```
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['country'].value_counts()
    for x_label, grp in _df_15.groupby('director')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('director')
_ = plt.ylabel('country')
```

```
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['rating'].value_counts()
    for x_label, grp in _df_16.groupby('country')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('country')
_ = plt.ylabel('rating')
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x79e670697190>
```

```
<string>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `y` variable to `hue` and set
`legend=False` for the same effect.
```

```
from matplotlib import pyplot as plt
import seaborn as sns
figsize = (12, 1.2 * len(_df_17['type'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(_df_17, x='release_year', y='type', inner='stick',
palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```

```
<string>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `y` variable to `hue` and set
`legend=False` for the same effect.
```

```
from matplotlib import pyplot as plt
import seaborn as sns
figsize = (12, 1.2 * len(_df_18['title'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(_df_18, x='release_year', y='title', inner='stick',
palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```

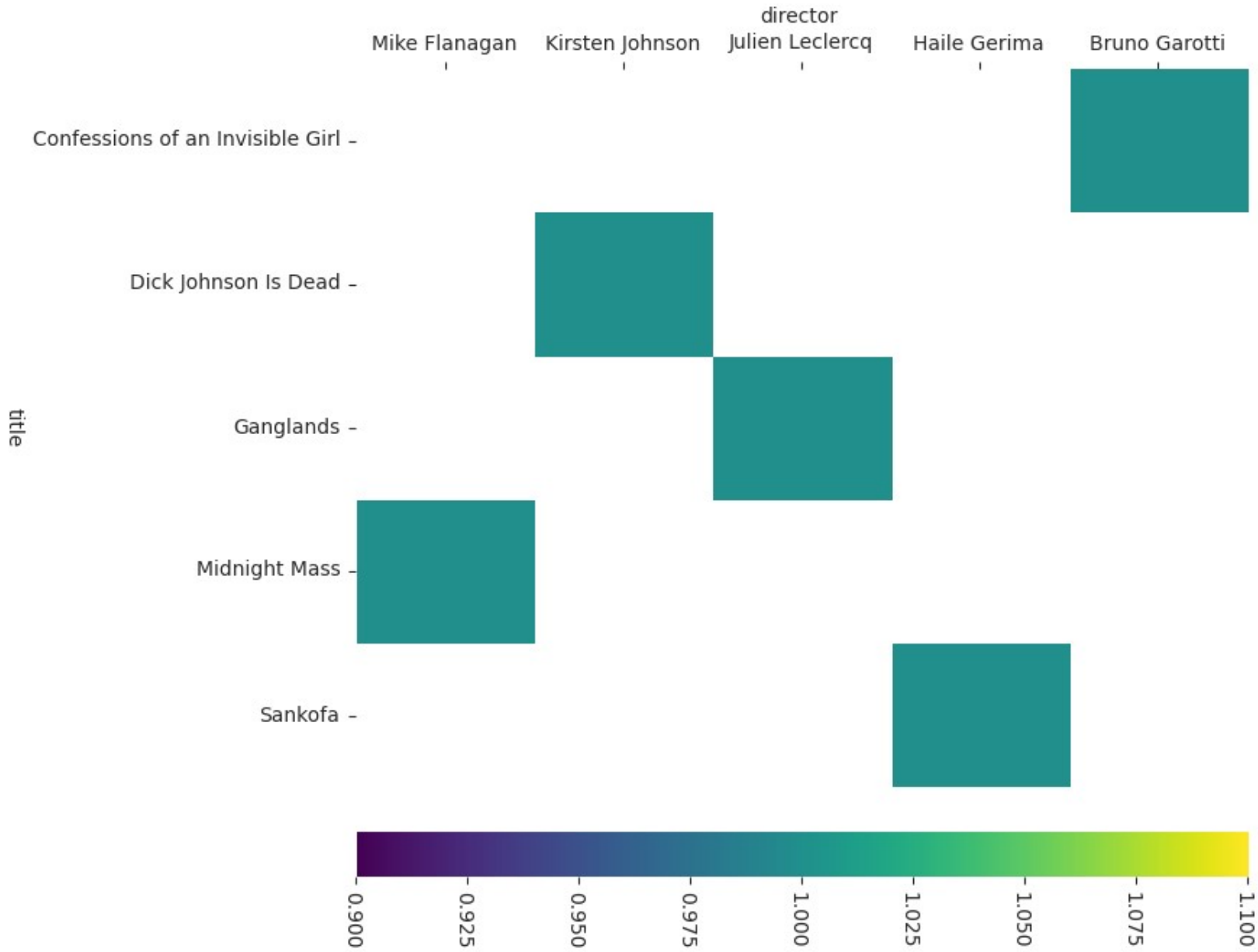
```
<string>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `y` variable to `hue` and set
`legend=False` for the same effect.
```

```
from matplotlib import pyplot as plt
import seaborn as sns
figsize = (12, 1.2 * len(_df_19['director'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(_df_19, x='release_year', y='director', inner='stick',
palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```

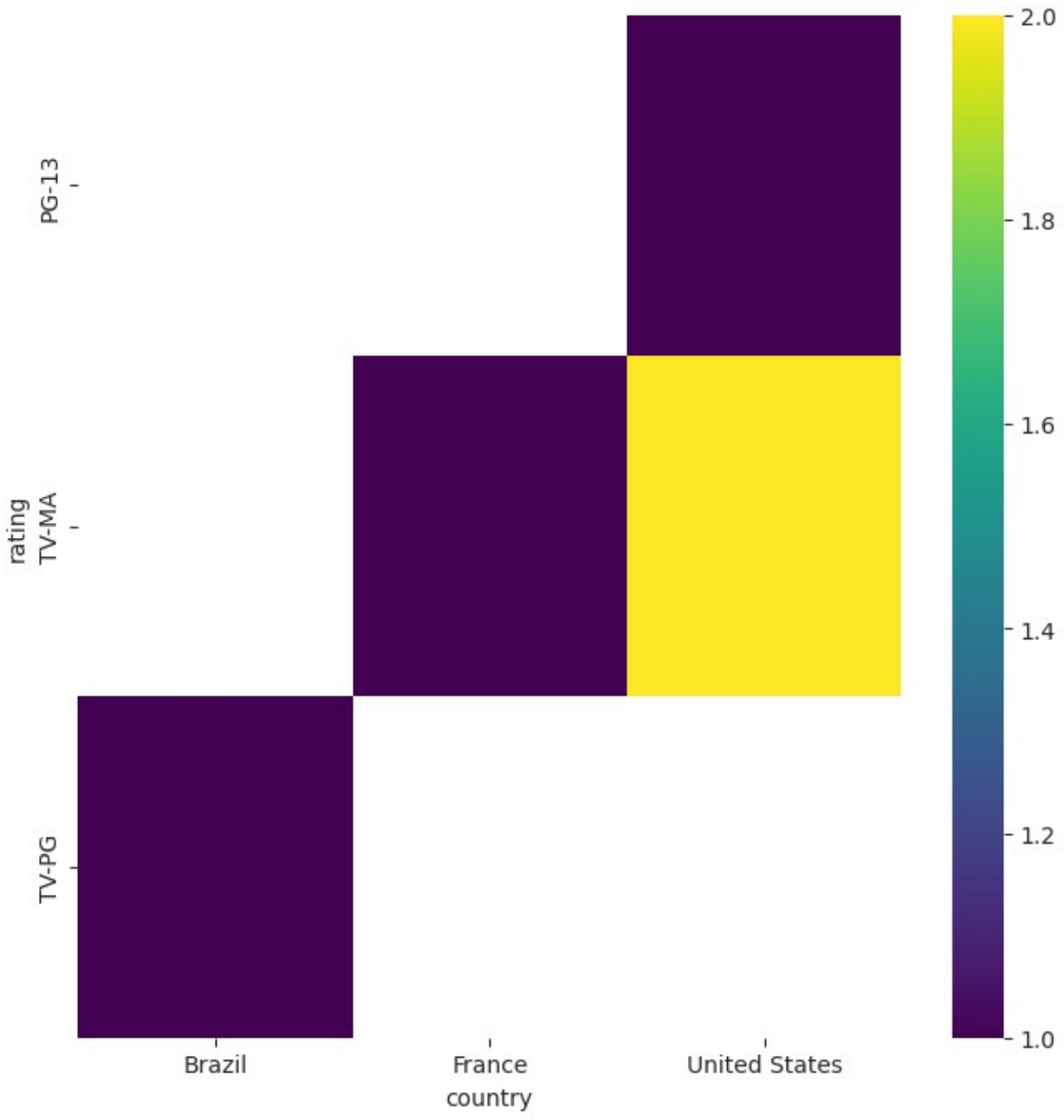
```
<string>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `y` variable to `hue` and set
`legend=False` for the same effect.
```

```
from matplotlib import pyplot as plt
import seaborn as sns
figsize = (12, 1.2 * len(_df_20['country'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(_df_20, x='release_year', y='country', inner='stick',
palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```

```
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['director'].value_counts()
    for x_label, grp in _df_14.groupby('title')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('title')
_ = plt.ylabel('director')
```


director
Mike Flanagan
Kirsten Johnson
Julien Leclercq
Haile Gerima
Bruno Garotti
title
Confessions of an Invisible Girl
Dick Johnson Is Dead
Ganglands
Midnight Mass
Sankofa
0.900
0.925
0.950
0.975
1.000
1.025
1.050
1.075
1.100

```
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['rating'].value_counts()
    for x_label, grp in _df_16.groupby('country')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('country')
_ = plt.ylabel('rating')
```


PG-13
TV-MA
TV-PG
rating
Brazil
France
United States
country
2.0
1.8
1.6
1.4
1.2
1.0