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

```
df=pd.read_csv('/content/netflix.csv')
df.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.shape# total rows and columns
```

```
(8807, 12)
```

```
df.info()# details of all columns
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

```
#handling null values
df['director']=df['director'].fillna('Unknown')
df['cast']=df['cast'].fillna('Unknown')
df['country']=df['country'].fillna('Unknown')
df['date_added']=df['date_added'].fillna('Unknown')
df['rating']=df['rating'].fillna('Unknown')
df['duration']=df['duration'].fillna('Unknown')
```

```
df.info() # null values handled
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
```

```
 3   director      8807 non-null   object
 4   cast          8807 non-null   object
 5   country       8807 non-null   object
 6   date_added    8807 non-null   object
 7   release_year  8807 non-null   int64
 8   rating        8807 non-null   object
 9   duration      8807 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

```
df.dtypes# column datatypes
```

	0
show_id	object
type	object
title	object
director	object
cast	object
country	object
date_added	object
release_year	int64
rating	object
duration	object
listed_in	object
description	object

dtype: object

```
df['release_year'].value_counts() # release year count
```

release_year	count
2018	1147
2017	1032
2019	1030
2020	953
2016	902
...	...
1959	1
1925	1
1961	1
1947	1
1966	1

74 rows × 1 columns

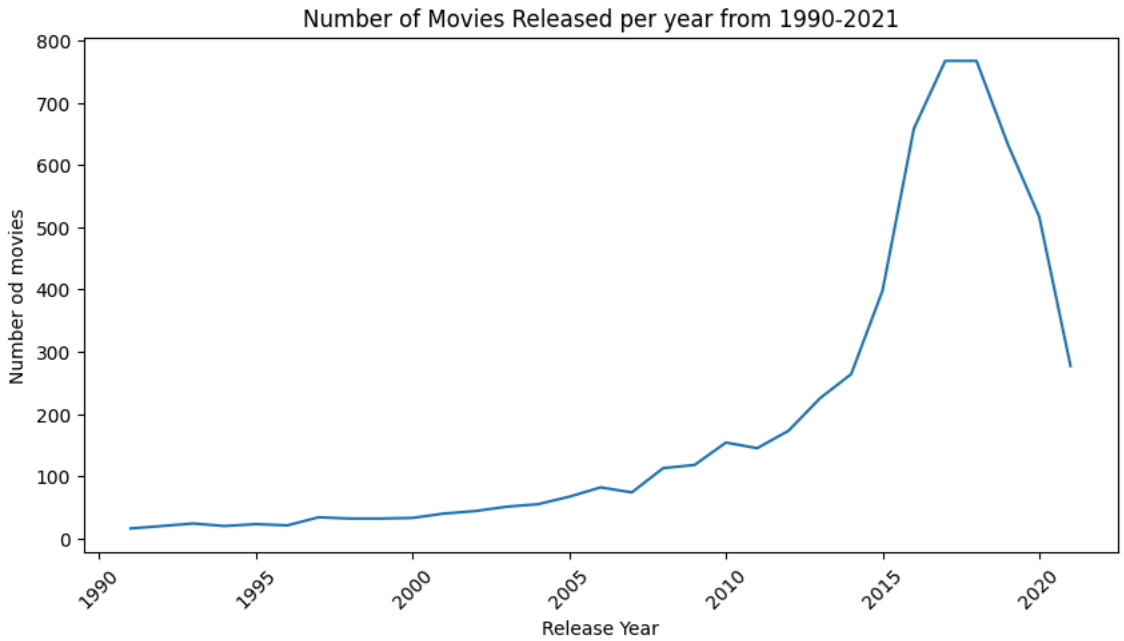
dtype: int64

```
# question 1 How has the number of movies released per year changed over the last 20-30 years?

#filtering onle movies rows
movies_data=df[df['type']=='Movie']
movies_last_30yearsdata=movies_data[movies_data['release_year']>=1991]


new_data=movies_last_30yearsdata.groupby('release_year')['show_id'].count()
new_data
```

release_year	show_id
1991	16
1992	20
1993	24
1994	20
1995	23
1996	21
1997	34
1998	32
1999	32
2000	33
2001	40
2002	44
2003	51
2004	55
2005	67
2006	82
2007	74
2008	113
2009	118
2010	154
2011	145
2012	173
2013	225
2014	264
2015	398
2016	658
2017	767
2018	767
2019	633
2020	517
2021	277

dtype: int64

```
# following plot helps to visualise the Number of Movies Released per year from 1990-2021
plt.figure(figsize=(10,5))
plt.plot(new_data.index,new_data.values)
plt.ylabel('Number of Movies')
plt.xlabel('Release Year')
plt.ylabel('Number od movies')
plt.title('Number of Movies Released per year from 1990-2021')
plt.xticks(rotation=45)

plt.show()
```



Insights

1. Steady Increase: There has been a consistent rise in the number of movies released each year since the early 1990s.
2. Surge in the 2000s: The early 2000s saw a notable spike in releases, likely influenced by the rise of digital technology and online distribution.

Recommendations

Here are some short recommendations based on the trends in movie releases:

1. **Explore Diverse Genres**: Consider watching films from various genres and countries to broaden your cinematic experience.
2. **Support Indie Films**: Look for independent films, which often offer unique storytelling and fresh perspectives.
3. **Engage with Film Communities**: Join online forums or local clubs to discuss and share recommendations with fellow film enthusiasts.

```
# question 2. Comparison of tv shows vs. movies.
type_wise_count=df['type'].value_counts()
print('tv shows vs. movies')
type_wise_count
```

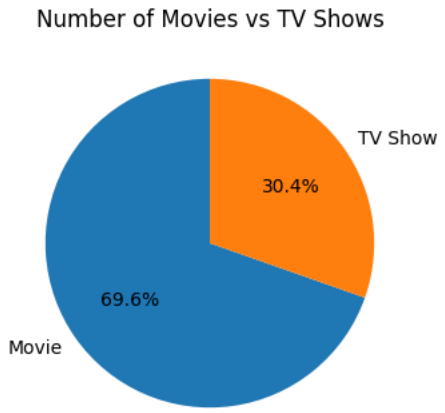
```
tv shows vs. movies
```

type	count
Movie	6131
TV Show	2676

dtype: int64

Number of Movies vs TV Shows

```
# @title Number of Movies vs TV Shows

# Create a pie chart
plt.figure(figsize=(8,4))
plt.pie(type_wise_count, labels=type_wise_count.index, autopct='%1.1f%%', startangle=90)
plt.title('Number of Movies vs TV Shows')
plt.show()
```



Insights

1. Volume Difference: movies outnumber TV shows.
2. Volume Difference: TV shows often outnumber movies, reflecting a growing trend toward episodic content and binge-watching culture.

recommendations

1. Diversify Your Watchlist: Mix genres and formats to keep your viewing experience fresh and engaging.
2. Explore International Content: Check out foreign films and series to discover new storytelling styles and cultural perspectives.
3. Binge Strategically: If you enjoy binge-watching, set aside time to fully immerse yourself in longer series without distractions.

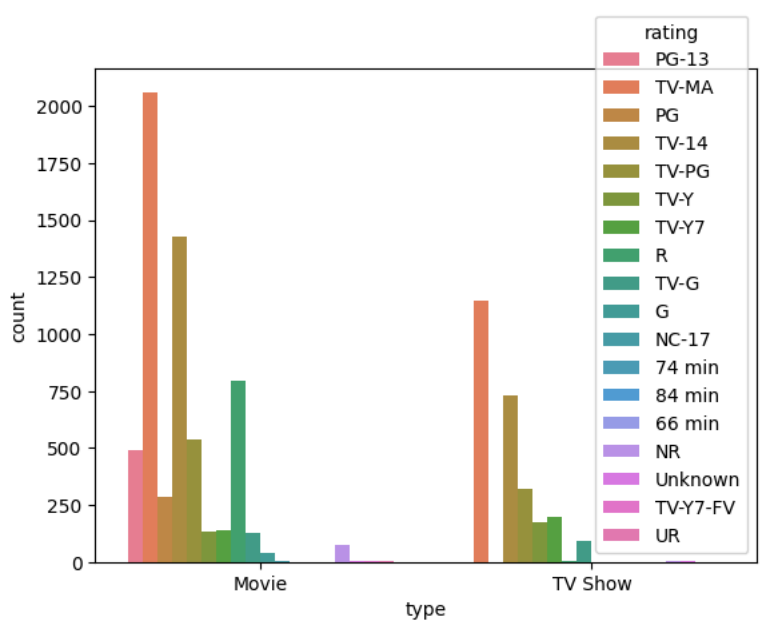
```
# rating count for movies and tv series
rating_count=df[['type','rating']].value_counts()
print('movies and tv shows rating count')
rating_count
```

```
movies and tv shows rating count
```

type	rating	count
Movie	TV-MA	2062
	TV-14	1427
TV Show	TV-MA	1145
Movie	R	797
TV Show	TV-14	733
Movie	TV-PG	540
	PG-13	490
TV Show	TV-PG	323
Movie	PG	287
TV Show	TV-Y7	195
	TV-Y	176
Movie	TV-Y7	139
	TV-Y	131
	TV-G	126
TV Show	TV-G	94
Movie	NR	75
	G	41
	TV-Y7-FV	5
TV Show	NR	5
Movie	UR	3
	NC-17	3
TV Show	Unknown	2
	R	2
Movie	Unknown	2
	74 min	1
	84 min	1
TV Show	TV-Y7-FV	1
Movie	66 min	1

dtype: int64

```
sns.countplot(x='type',hue='rating',data=df)
plt.show()
```



Insights

1. Understand the Distribution: The output will show you how many movies and TV shows exist within each rating category (e.g., PG, R, TV-MA).
2. Insights on Ratings:

A higher count of certain ratings could indicate a target demographic or content strategy. The presence of various ratings can reflect the diversity of content available.

Recommendations

1. Check Ratings Before Watching: Always look at ratings to find content that aligns with your preferences and comfort levels.
2. Explore Different Ratings: Don't shy away from higher-rated content; it may offer compelling storytelling that you wouldn't normally consider.
3. Utilize Parental Controls: If watching with family, use parental controls to ensure age-appropriate content for younger viewers.
4. Experiment with Ratings: Try watching shows or movies outside your usual ratings to discover new favorites.

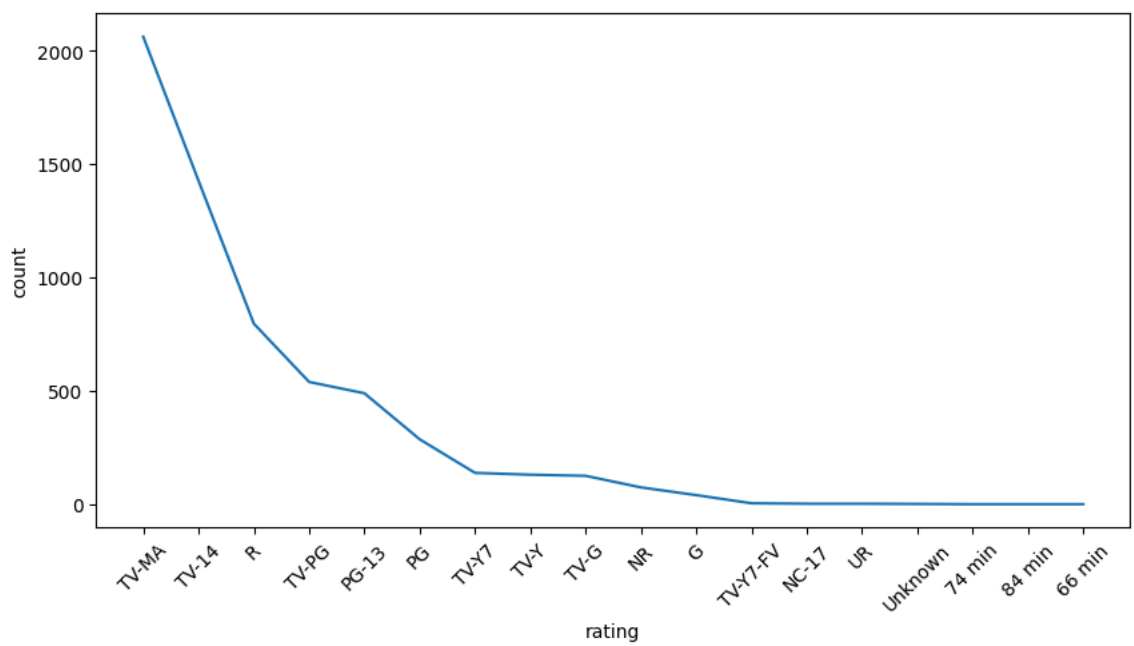
```
# movies rating count
movies_data=df[df['type']=='Movie']
print('movies rating count')
movies_rating_count=movies_data['rating'].value_counts().reset_index()
movies_rating_count
```

```
movies_rating_count
```

	rating	count
0	TV-MA	2062
1	TV-14	1427
2	R	797
3	TV-PG	540
4	PG-13	490
5	PG	287
6	TV-Y7	139
7	TV-Y	131
8	TV-G	126
9	NR	75
10	G	41
11	TV-Y7-FV	5
12	NC-17	3
13	UR	3
14	Unknown	2
15	74 min	1
16	84 min	1
17	66 min	1

Next steps: Generate code with `movies_rating_count` | View recommended plots | New interactive sheet

```
plt.figure(figsize=(10,5))
sns.lineplot(x='rating',y='count',data=movies_rating_count)
plt.xticks(rotation=45)
plt.show()
```



Insoghts

1. Viewer Satisfaction: Consistently high ratings can highlight successful genres or filmmakers.
2. Market Trends: Shifts in rating distributions over time could reflect changing audience tastes or external influences.
3. Recommendation Potential: High-rated movies can serve as strong candidates for recommendations to enhance viewer engagement.

recommendations

1. User Engagement: Encourage viewers to rate movies after watching, increasing the volume of feedback and improving future analyses.
2. Thematic Promotions: Leverage themes from high-rated movies (e.g., genres, directors) for special promotions or events.

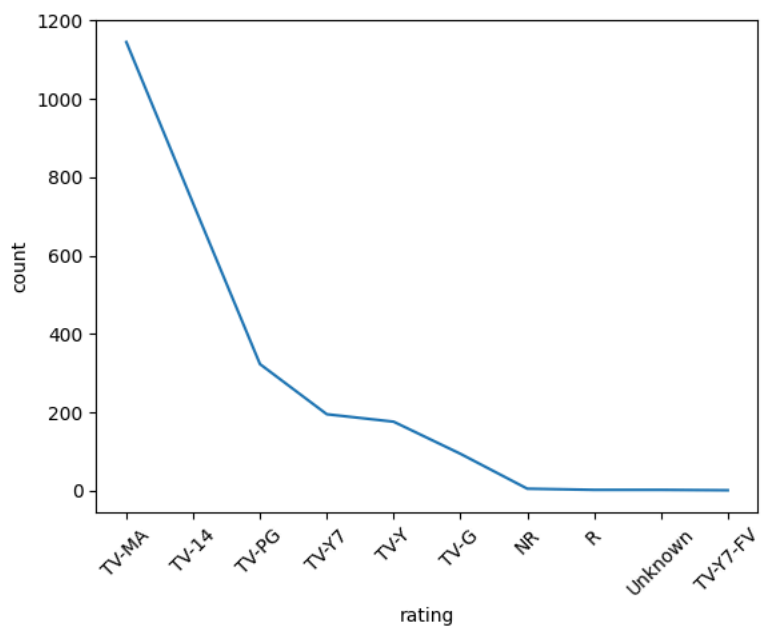
```
# Tv shows rating count
tv_shows_data=df[df['type']=='TV Show']
print('tv shows rating count')
tv_shows_rating_count=tv_shows_data['rating'].value_counts().reset_index()
tv_shows_rating_count
```

```
tv shows rating count
```

	rating	count
0	TV-MA	1145
1	TV-14	733
2	TV-PG	323
3	TV-Y7	195
4	TV-Y	176
5	TV-G	94
6	NR	5
7	R	2
8	Unknown	2
9	TV-Y7-FV	1

Next steps: Generate code with tv_shows_rating_count | View recommended plots | New interactive sheet

```
sns.lineplot(x='rating',y='count',data=tv_shows_rating_count)
plt.xticks(rotation=45)
plt.show()
```



insights

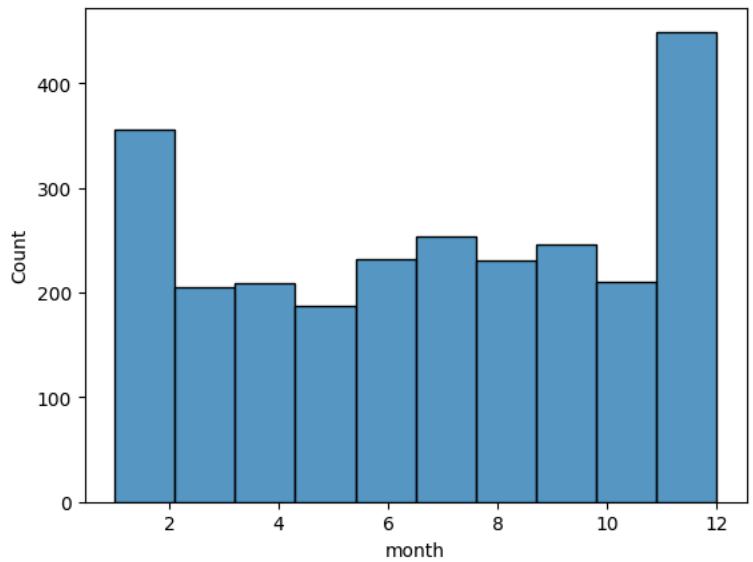
1. netflix prefers TV-MA,TV-14,TV-PG shows among all other TV shows.
2. netflix added least number of TV shows of rating TV-Y7_FV.

Recommendations

1. Targeted Marketing: Use ratings data to tailor marketing campaigns for specific genres that perform well.
2. Monitor Trends: Regularly analyze rating shifts to adapt programming strategies and identify new trends early.

```
#What is the best time to launch a TV show?
tv_shows_data=df[df['type']=='TV Show']


#extracting month from date_added column
tv_shows_data['date_added'] = pd.to_datetime(df['date_added'],errors='coerce')
tv_shows_data['month']=tv_shows_data['date_added'].dt.month


Total_Tv_shows_added_per_month=tv_shows_data['month'].value_counts().sort_index()
Total_Tv_shows_in_netflix=tv_shows_data['show_id'].count()


print('Total Tv shows added in a month')
print(Total_Tv_shows_added_per_month)

print('Total Tv shows in netflix')
print(Total_Tv_shows_in_netflix)

sns.histplot(x='month',data=tv_shows_data,bins=10)
plt.show()
```

```
<ipython-input-353-5c6fcb06b4fc>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  tv_shows_data['date_added'] = pd.to_datetime(df['date_added'],errors='coerce')
<ipython-input-353-5c6fcb06b4fc>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  tv_shows_data['month']=tv_shows_data['date_added'].dt.month
Total Tv shows added in a month
month
1.0     181
2.0     175
3.0     205
4.0     209
5.0     187
6.0     232
7.0     254
8.0     230
9.0     246
10.0    210
11.0    199
12.0    250
Name: count, dtype: int64
Total Tv shows in netflix
2676
```



Insights

1. july, december,september are the top three months in which more TV shows are added on netflix platform.
2. during july(7.0) netflix has added most number of TV shows.
3. during december(12.0) month netflix has added more TV shows after july month.
4. in february(2.0) least number of TV shows has been added on netflix.

Recommandtions

1. Align with Peak Months: Launch new shows during months with high historical additions to capture viewer interest.
2. Seasonal Themes: Plan launches around holidays or seasonal events to enhance relevance and viewer engagement.
3. Monitor Audience Engagement: Track viewer responses post-launch to refine future release strategies and optimize timing.

```
# Analysis of actors/directors of different types of shows/movies.
```

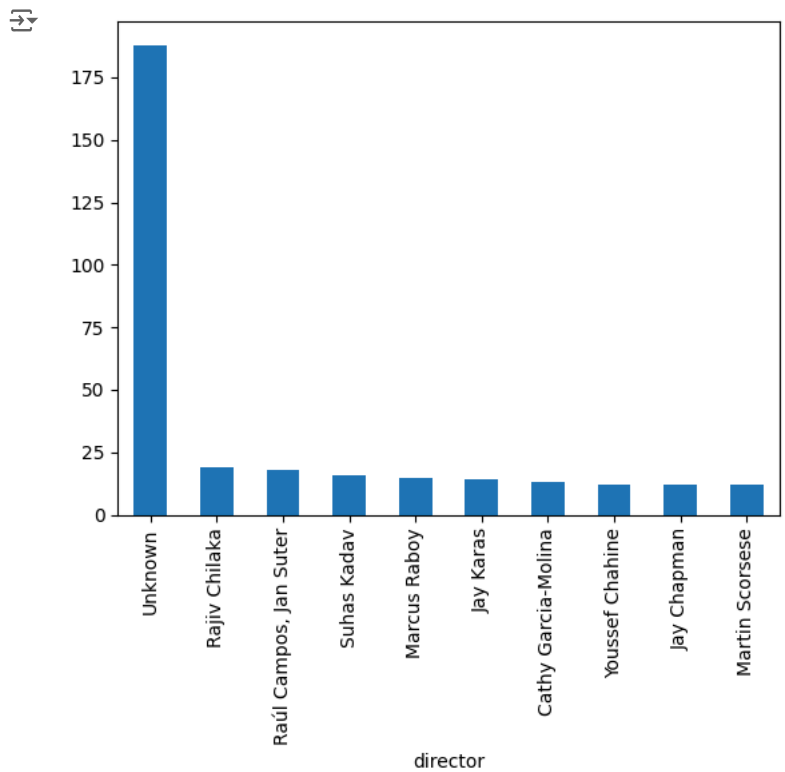
```
print('Top 10 movies director')
Top_10_movies_director=movies_data['director'].value_counts().sort_values(ascending=False).head(10)
Top_10_movies_director
```

```
Top 10 movies director
```

director	count
Unknown	188
Rajiv Chilaka	19
Raúl Campos, Jan Suter	18
Suhas Kadav	16
Marcus Raboy	15
Jay Karas	14
Cathy Garcia-Molina	13
Youssef Chahine	12
Jay Chapman	12
Martin Scorsese	12

dtype: int64

```
Top_10_movies_director.plot(kind='bar')
plt.show()
```



Insights

1. Missing values in the director column are filled with the string 'Unknown'
2. Rajiv Chilaka has directed highest movies among all other directors.
3. Rajiv Chilaka, Raúl Campos and Jan Suter are the top 3 directors on directoring highest number of movies across all the years.
4. Youssef Chahine, Jay Chapman and Martin Scorsese have directed same number of movies across the years which is least among all other directors.

Recommendations

1. Highlight Top Talent: Promote works by the most popular directors and actors to attract viewers, creating themed collections or features.
2. Explore Genre-Specific Talent: Identify directors and actors who excel in particular genres and tailor marketing campaigns to highlight their work.
3. Foster Collaborations: Encourage partnerships between successful directors and actors to generate buzz and enhance the quality of new projects.

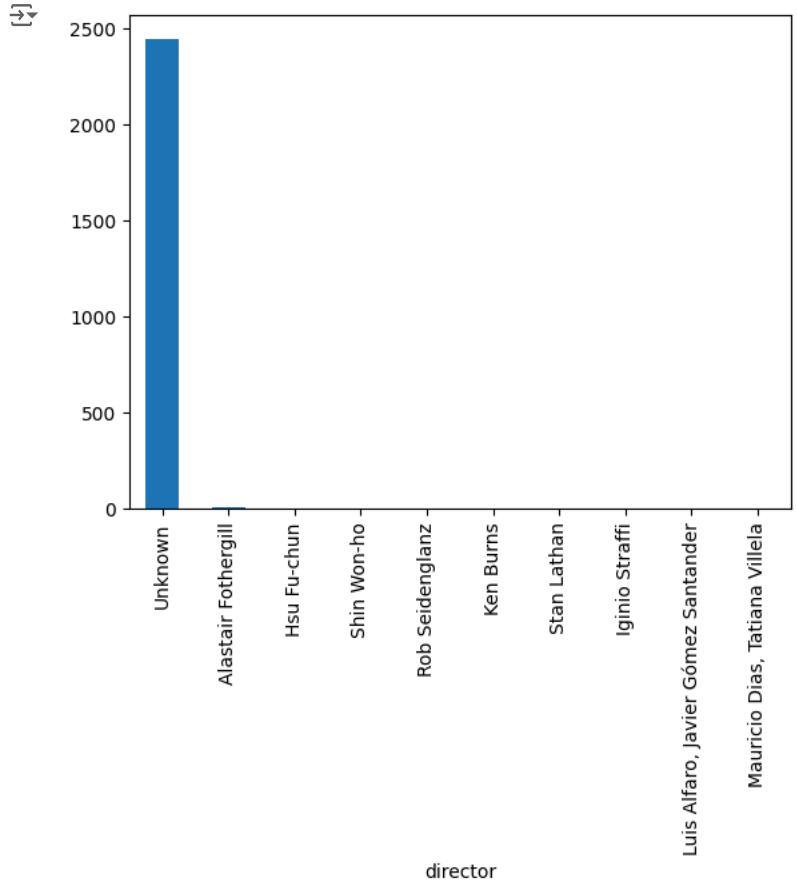
```
print('Top 10 TV shows directors')
Top_10_TV_shows_directors=tv_shows_data['director'].value_counts().sort_values(ascending=False).head(10)
Top_10_TV_shows_directors
```

```
Top 10 TV shows directors
```

director	count
Unknown	2446
Alastair Fothergill	3
Hsu Fu-chun	2
Shin Won-ho	2
Rob Seidenglanz	2
Ken Burns	2
Stan Lathan	2
Iginio Straffi	2
Luis Alfaro, Javier Gómez Santander	1
Mauricio Dias, Tatiana Villela	1

dtype: int64

```
Top_10_TV_shows_directors.plot(kind='bar')
plt.show()
```



Insights

1. Missing values are filled with the string 'Unknown'.
2. Alastair Fothergill has directed highest Tv shows than other directors.
3. Alastair Fothergill,Hsu Fu-chun,Shin Won-ho have directed more Tv shows than other directors.

Recommondations

1.Your code to analyze the top directors of TV shows is straightforward and effective! Here's how to refine it a bit for clarity and structure:

Code

```
# Analyzing top 10 TV show directors
print('Top 10 TV Shows Directors:')
Top_10_TV_shows_directors = tv_shows_data['director'].value_counts().sort_values(ascending=False).head(10)
print(Top_10_TV_shows_directors)
```

Short Insights

1. Director Popularity: Identifying the top directors helps highlight those shaping trends in TV programming.
2. Genre Specialization: Some directors may excel in specific genres, indicating preferences in storytelling styles among viewers.
3. Collaborative Success: Frequent collaborations among directors and actors may enhance the quality and popularity of shows.

Recommendations

1. Promote Top Directors: Create marketing campaigns that spotlight the works of these directors to attract their fan base.
2. Explore Genre Connections: Investigate how these directors influence different genres, tailoring content suggestions accordingly.
3. Support New Directors: Highlight emerging directors alongside established names to diversify the content offering.
4. Fan Engagement: Encourage audience interaction with content related to popular directors, such as Q&A sessions or behind-the-scenes features.

```
print('top 10 movie actors')
movies_data['cast'].str.split(',').explode().value_counts().head(10)
```

```
top 10 movie actors
```

cast	count
Unknown	475
Anupam Kher	38
Rupa Bhimani	27
Om Puri	27
Shah Rukh Khan	26
Boman Irani	25
Paresh Rawal	25
Julie Tejwani	24
Akshay Kumar	23
Rajesh Kava	21

dtype: int64

```
print('top 10 cast for tv shows')
df['cast'].str.split(',').explode().value_counts().head(10)
```

```
top 10 cast for tv shows
```

cast	count
Unknown	825
Anupam Kher	39
Rupa Bhimani	31
Takahiro Sakurai	30
Julie Tejwani	28
Om Puri	27
Shah Rukh Khan	26
Rajesh Kava	26
Andrea Libman	25
Paresh Rawal	25

dtype: int64

Insights

1. Anupam Kher has topped in the list of cast in Tv shows.He has acted in total 39 TV shows.
2. Rupa Bhimani and Takahiro Sakurai both have acted in 30 or more movies who are 2 nd and 3rd in the list.
3. Cast Popularity: The top cast members can indicate which actors are currently resonating with audiences in TV shows.

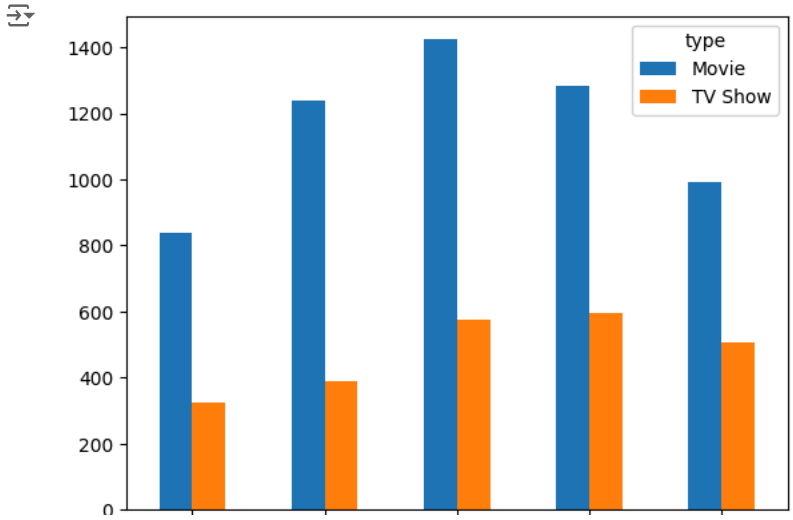
Recommendations

1. Promotional Campaigns: Leverage the popularity of top cast members in marketing efforts to attract their existing fan bases and boost viewership.
2. Diverse Casting: While promoting well-known actors, also highlight emerging talent to diversify your content and appeal to a broader audience.

```
# Does Netflix has more focus on TV Shows than movies in recent years

  # considering recent 5 years data as recent data

df['year_added']=pd.to_datetime(df['date_added'],errors='coerce').dt.year
max_year_added=df['year_added'].max()
recent_data=df[df['year_added']>=max_year_added-4]
focus_data=recent_data.groupby(['year_added','type'])['show_id'].count().unstack()
print(focus_data)

Total_movies=focus_data['Movie'].sum()
Total_tv_shows=focus_data['TV Show'].sum()

print('Total_movies',Total_movies)
print('Total_tv_shows',Total_tv_shows)
```

```
type        Movie  TV Show
year_added
2017.0        839      325
2018.0       1237      388
2019.0       1424      575
2020.0       1284      594
2021.0        993      505
Total_movies 5777
Total_tv_shows 2387
```

```
focus_data.plot(kind='bar')
plt.show()
```



Insights

1. in 2021 the number of movies and Tv shows released are less compare to previous year 2020.
2. averagely every year 1206 movies have added on the netflix and for tvs shows it is 477.4

Recommendations

1. Expand Original Series: Increase investment in original series to capitalize on viewer demand for binge-worthy content, potentially enhancing subscriber retention.