**Group no : 04**

**204 Aryan Meshram**

**210 Shreya Borle**

**212 Snehal Chavan**

**Dataset : Netflix**

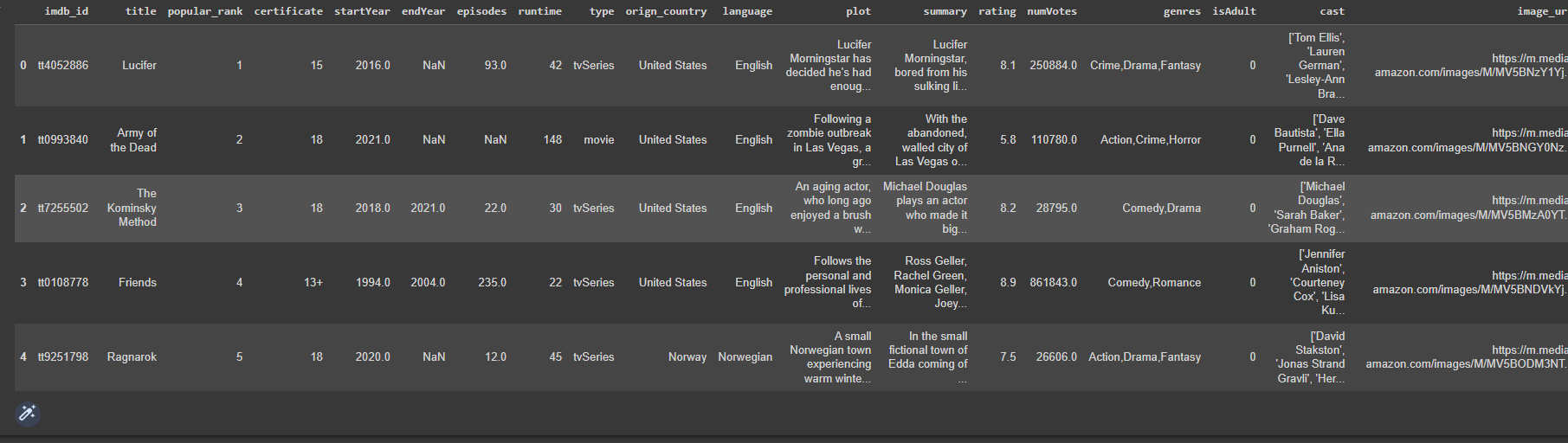
**import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

**df = pd.read\_csv('/content/netflix\_list.csv')**

**df.head()**



#1) df[df.duplicated()]



#2) df.runtime[(df.startYear == 2022) & (df.type != 'movie')].head(20)

1199 \N

1214 \N

1627 \N

3023 \N

3086 \N

3248 \N

3837 \N

3849 \N

4133 \N

4377 \N

4520 \N

4721 22

5063 \N

5081 \N

5255 \N

5357 \N

5490 \N

5575 \N

5664 \N

5756 7

Name: runtime, dtype: object

df.dtypes

#3) missing\_values  = df.isnull().sum()

df['startYear'] = df['startYear'].fillna('Unknown')

df['episodes'] = df['episodes'].fillna('No Data')

df['certificate'] = df['certificate'].fillna('No certificate')

df['numVotes'] = df['numVotes'].fillna('No rate')

df['rating'] = df['rating'].fillna('No rate')

df['plot'] = df['certificate'].fillna('No Data')

df['language'] = df['language'].fillna('Unknown')

df['genres'] = df['genres'].fillna('No Genre')

df['type'] = df['type'].fillna('No Type')

df['runtime'] = df['runtime'].fillna('Unknown')

#4) Calculate the sizes

movies = df.loc[df['type'].isin(['movie', 'short', 'tvMovie', 'video', 'videoGame', 'tvShort'])].shape[0]

tv\_shows = df.loc[df['type'].isin(['tvSeries', 'tvEpisode', 'tvSpecial', 'tvMiniSeries'])].shape[0]

# Define the labels and colors

labels = ['Movies', 'TV Shows']

sizes = [movies, tv\_shows]

colors = ['#ff9999', '#abcdef']  # Custom colors for the pie slices

#5) Create the pie chart

plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90, shadow=True)

#6) Customize the chart appearance

plt.title('Proportion of Movies and TV Shows')

plt.axis('equal')  # Ensure the pie chart is circular

#7) Add a legend

plt.legend(loc='upper right')

# Show the chart

plt.show()

 #8) Filter and aggregate the data

# Filter out rows where the 'rating' column is 'No rate'

df.rating = df.rating[df.rating != 'No rate']

# Filter out rows where the 'numVotes' column is 'No rate'

df.numVotes = df.numVotes[df.numVotes != 'No rate']

# Filter out rows where the 'startYear' column is 'Unknown'

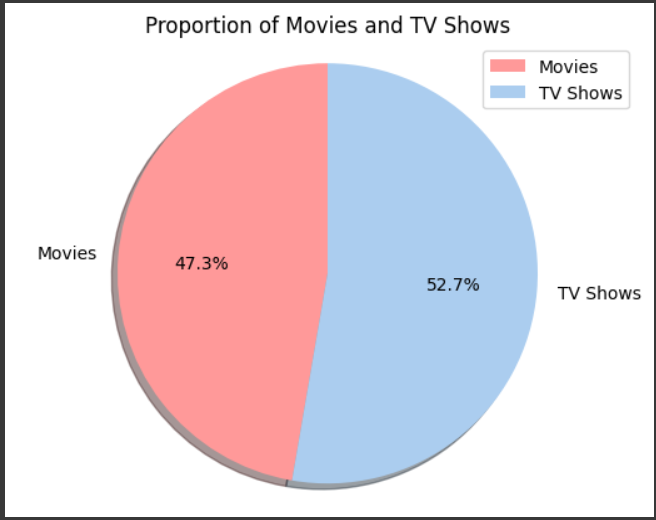
df.startYear = df.startYear[df.startYear != 'Unknown']

# Group the filtered data by 'startYear' and calculate the mean of 'rating' and the sum of 'numVotes'

rate\_per\_year = df.groupby('startYear').agg({'rating':'mean','numVotes':'sum'})

# Select just the last 15 years until 2021

rate\_per\_year = rate\_per\_year.iloc[:-1].tail(15)



#9) Filter and aggregate the data

# Filter out rows where the 'rating' column is 'No rate'

df.rating = df.rating[df.rating != 'No rate']

# Filter out rows where the 'numVotes' column is 'No rate'

df.numVotes = df.numVotes[df.numVotes != 'No rate']

# Filter out rows where the 'startYear' column is 'Unknown'

df.startYear = df.startYear[df.startYear != 'Unknown']

# Group the filtered data by 'startYear' and calculate the mean of 'rating' and the sum of 'numVotes'

rate\_per\_year = df.groupby('startYear').agg({'rating':'mean','numVotes':'sum'})

# Select just the last 15 years until 2021

rate\_per\_year = rate\_per\_year.iloc[:-1].tail(15)

# Create the figure object and plot the data

fig, ax1 = plt.subplots(figsize=(11, 6))

# Plot the 'rating' column as a line chart with label 'Rating'

ax1.plot(rate\_per\_year['rating'], label='Rating', color='#852852', marker='o', linestyle='-', linewidth=2)

# Set the y-axis label for the line chart

ax1.set\_ylabel('Rating')

# Create a second y-axis for the bar chart

ax2 = ax1.twinx()

# Plot the 'numVotes' column as a bar chart with label 'Number of Votes'

ax2.bar(rate\_per\_year.index, rate\_per\_year['numVotes'], label='Number of Votes', color='skyblue', alpha=0.7)

# Set the y-axis label for the bar chart

ax2.set\_ylabel('Number of Votes')

# Set x-axis tick labels to every other index from rate\_per\_year

ax1.set\_xticks(rate\_per\_year.index)

ax1.set\_xticklabels(rate\_per\_year.index.astype(int), rotation=45)

# Add a legend to the plot

lines, labels = ax1.get\_legend\_handles\_labels()

bars, bar\_labels = ax2.get\_legend\_handles\_labels()

ax1.legend(lines + bars, labels + bar\_labels, loc='upper right')

# Add a title

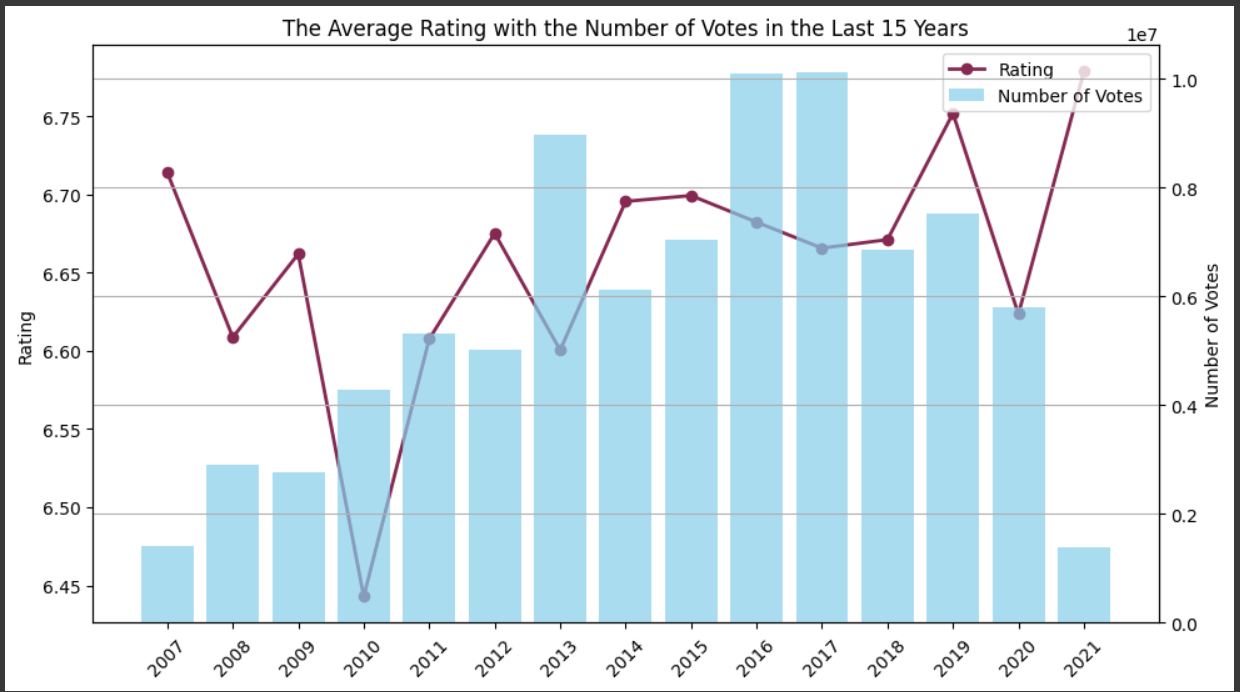
plt.title("The Average Rating with the Number of Votes in the Last 15 Years")

# Add grid lines

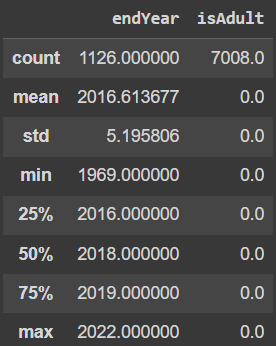
plt.grid(True)

# Show the plot

plt.show()



10) df.describe()



11) df[df.endYear.isnull()]



#12) Find maximum number of votes

vote=df['numVotes'].max()

print(" maximum number of votes:",vote)

maximum number of votes: 1697849.0

#13) Top ten series

print("Top ten series are:",df.iloc[1:11])

Top ten series are: imdb\_id title popular\_rank certificate startYear \

1 tt0993840 Army of the Dead 2 18 2021.0

2 tt7255502 The Kominsky Method 3 18 2018.0

3 tt0108778 Friends 4 13+ 1994.0

4 tt9251798 Ragnarok 5 18 2020.0

5 tt5028002 StartUp 6 18 2016.0

6 tt0413573 Grey's Anatomy 7 15+ 2005.0

7 tt12809988 Sweet Tooth 8 16 2021.0

8 tt2741602 The Blacklist 9 16+ 2013.0

9 tt5774002 Jupiter's Legacy 10 18 2021.0

10 tt7945720 Dirty John 11 16 2018.0

endYear episodes runtime type orign\_country language plot \

1 NaN No Data 148 movie United States English 18

2 2021.0 22.0 30 tvSeries United States English 18

3 2004.0 235.0 22 tvSeries United States English 13+

4 NaN 12.0 45 tvSeries Norway Norwegian 18

5 2018.0 30.0 44 tvSeries United States English 18

6 NaN 381.0 41 tvSeries United States English 15+

7 NaN 8.0 \N tvSeries United States English 16

8 NaN 175.0 43 tvSeries United States English 16+

9 2021.0 8.0 56 tvSeries United States English 18

10 NaN 16.0 44 tvSeries United States English 16

summary rating numVotes \

1 With the abandoned, walled city of Las Vegas o... 5.8 110780.0

2 Michael Douglas plays an actor who made it big... 8.2 28795.0

3 Ross Geller, Rachel Green, Monica Geller, Joey... 8.9 861843.0

4 In the small fictional town of Edda coming of ... 7.5 26606.0

5 Miami - A desperate banker needs to conceal st... 8.0 16980.0

6 A medical based drama centered around Meredith... 7.5 260703.0

7 A boy who is half human and half deer survives... 8.2 9622.0

8 A highly articulate, erudite and intelligent b... 8.0 207174.0

9 The first generation of superheroes kept the w... 6.8 27309.0

10 Debra Newell (Connie Britton) has a seemingly ... 7.2 16578.0

genres isAdult \

1 Action,Crime,Horror 0

2 Comedy,Drama 0

3 Comedy,Romance 0

4 Action,Drama,Fantasy 0

5 Crime,Thriller 0

6 Drama,Romance 0

7 Action,Adventure,Drama 0

8 Crime,Drama,Mystery 0

9 Action,Adventure,Drama 0

10 Crime,Drama 0

cast \

1 ['Dave Bautista', 'Ella Purnell', 'Ana de la R...

2 ['Michael Douglas', 'Sarah Baker', 'Graham Rog...

3 ['Jennifer Aniston', 'Courteney Cox', 'Lisa Ku...

4 ['David Stakston', 'Jonas Strand Gravli', 'Her...

5 ['Adam Brody', 'Edi Gathegi', 'Otmara Marrero'...

6 ['Ellen Pompeo', 'Chandra Wilson', 'James Pick...

7 ['Nonso Anozie', 'Christian Convery', 'Stefani...

8 ['James Spader', 'Megan Boone', 'Diego Klatten...

9 ['Josh Duhamel', 'Ben Daniels', 'Leslie Bibb',...

10 ['Connie Britton', 'Christian Slater', 'Eric B...

image\_url

1 <https://m.media-amazon.com/images/M/MV5BNGY0Nz>...

2 <https://m.media-amazon.com/images/M/MV5BMzA0YT>...

3 <https://m.media-amazon.com/images/M/MV5BNDVkYj>...

4 <https://m.media-amazon.com/images/M/MV5BODM3NT>...

5 <https://m.media-amazon.com/images/M/MV5BMTAxNT>...

6 <https://m.media-amazon.com/images/M/MV5BMjgwNG>...

7 <https://m.media-amazon.com/images/M/MV5BOTk4ZD>...

8 <https://m.media-amazon.com/images/M/MV5BZDA1Mz>...

9 <https://m.media-amazon.com/images/M/MV5BMDU4MW>...

10 [https://m.media-amazon.com/images/M/MV5BNmJhYT](https://m.media-amazon.com/images/M/MV5BNmJhYT" \t "_blank)...

14) Find the series which are ongoing

ongoing=df['endYear']

print("The ongoing series are:",df.isnull())

The ongoing series are: imdb\_id title popular\_rank certificate startYear endYear episodes \

0 False False False False False True False

1 False False False False False True False

2 False False False False False False False

3 False False False False False False False

4 False False False False False True False

... ... ... ... ... ... ... ...

7003 False False False False False True False

7004 False False False False False True False

7005 False False False False False True False

7006 False False False False False True False

7007 False False False False False True False

runtime type orign\_country language plot summary rating \

0 False False False False False False False

1 False False False False False False False

2 False False False False False False False

3 False False False False False False False

4 False False False False False False False

... ... ... ... ... ... ... ...

7003 False False False False False False False

7004 False False False False False False True

7005 False False False False False False False

7006 False False False False False False False

7007 False False False False False False False

numVotes genres isAdult cast image\_url

0 False False False False False

1 False False False False False

2 False False False False False

3 False False False False False

4 False False False False False

... ... ... ... ... ...

7003 False False False False False

7004 True False False False False

7005 False False False False False

7006 False False False False False

7007 False False False False False

[7008 rows x 19 columns]

#15) Print the summary of given dataset

Summary\_status = df.describe()

16) what are the countries who distributed more films & Movies ?

df.orign\_country.value\_counts()

United States 2836 - 551 United Kingdom 508 Japan 406 South Korea 316 ... Cyprus 1 Bahamas 1 Croatia 1 Puerto Rico 1 Haiti 1 Name: orign\_country, Length: 82, dtype: int64

#17) display mean of number of voters overall

print("Mean number of voters overall is:",df['numVotes'].mean())

Mean number of voters overall is: 19617.784833333335

#18) to check duplicate data

netflix[netflix.duplicated()]

#19) how many movies and tv shows of same genre?

netflix.genres.value\_counts().head(20)

Comedy 713 Drama 448 Documentary 431 Action,Adventure,Animation 253 Comedy,Drama 193 Drama,Romance 164 Adventure,Animation,Comedy 149 Crime,Drama,Mystery 145 Comedy,Drama,Romance 135 Action,Crime,Drama 133 Comedy,Romance 121 Reality-TV 118 Crime,Drama,Thriller 101 \N 87 Action,Adventure,Drama 87 Drama,Thriller 85 Crime,Drama 74 Comedy,Documentary 73 Crime,Documentary 69 Thriller 65

#20)  know the data type for each column?

netflix.dtypes

imdb\_id object title object popular\_rank object certificate object startYear float64 endYear float64 episodes float64 runtime object type object orign\_country object language object plot object summary object rating float64 numVotes float64 genres object isAdult int64 cast object image\_url object dtype: object

# Calculate the sizes

movies = df.loc[df['type'].isin(['movie', 'short', 'tvMovie', 'video', 'videoGame', 'tvShort'])].shape[0]

tv\_shows = df.loc[df['type'].isin(['tvSeries', 'tvEpisode', 'tvSpecial', 'tvMiniSeries'])].shape[0]

# Define the labels and colors

labels = ['Movies', 'TV Shows']

sizes = [movies, tv\_shows]

colors = ['#ff9999', '#abcdef']  # Custom colors for the pie slices

# Filter out rows where the 'rating' column is 'No rate'

df.rating = df.rating[df.rating != 'No rate']

# Filter out rows where the 'numVotes' column is 'No rate'

df.numVotes = df.numVotes[df.numVotes != 'No rate']

# Filter out rows where the 'startYear' column is 'Unknown'

df.startYear = df.startYear[df.startYear != 'Unknown']

# Group the filtered data by 'startYear' and calculate the mean of 'rating' and the sum of 'numVotes'

rate\_per\_year = df.groupby('startYear').agg({'rating':'mean','numVotes':'sum'})

# Select just the last 15 years until 2021

rate\_per\_year = rate\_per\_year.iloc[:-1].tail(15)

# Read in the Netflix code dataset

netflix= pd.read\_csv('/content/netflix\_list.csv')

#21) Check for missing values

print('Number of missing values in the dataset:', netflix.isnull().sum().sum())

Number of missing values in the dataset: 18121

# Remove rows with missing values

netflix = netflix.dropna()

#22) Check for duplicated rows

print('Number of duplicated rows in the dataset:', netflix.duplicated().sum())

Number of duplicated rows in the dataset: 0

# Calculate the mean rating for each category

mean\_ratings = netflix.groupby('type')['rating'].mean()

#23) Print the top 10 categories by mean rating

print('Top 10 categories by mean rating:')

print(mean\_ratings.nlargest(10))

Top 10 categories by mean rating:

type

tvSeries 7.619205

tvMiniSeries 7.416667

Name: rating, dtype: float64