mid-term1-jaamie-maarsh-joy-martin

October 12, 2023

[1]:
```
# importing all the necessary libraries into the workspace
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

#This command is to ignore all the warnings
warnings.filterwarnings("ignore")

# loading/reading of the datasets into a dataframe (df)
df_netflix= pd.read_csv('/Users/jaamiemaarshj/Desktop/  DAE Course Materials/
 ↪Computization and Visualisation/Mid term/netflix_titles.csv',␣
 ↪low_memory=False)

display(df_netflix.head())
df_netflix.info()
```

```
  show_id     type                  title         director  \
0      s1    Movie   Dick Johnson Is Dead  Kirsten Johnson
1      s2  TV Show          Blood & Water              NaN
2      s3  TV Show              Ganglands  Julien Leclercq
3      s4  TV Show  Jailbirds New Orleans              NaN
4      s5  TV Show           Kota Factory              NaN

                                                cast        country  \
0                                                NaN  United States
1  Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban…   South Africa
2  Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi…            NaN
3                                                NaN            NaN
4  Mayur More, Jitendra Kumar, Ranjan Raj, Alam K…          India

          date_added  release_year rating   duration  \
0  September 25, 2021          2020  PG-13     90 min
1  September 24, 2021          2021  TV-MA  2 Seasons
2  September 24, 2021          2021  TV-MA   1 Season
3  September 24, 2021          2021  TV-MA   1 Season
4  September 24, 2021          2021  TV-MA  2 Seasons
```

```
                                          listed_in  \
0                                    Documentaries
1    International TV Shows, TV Dramas, TV Mysteries
2  Crime TV Shows, International TV Shows, TV Act…
3                              Docuseries, Reality TV
4  International TV Shows, Romantic TV Shows, TV …

                                         description
0  As her father nears the end of his life, filmm…
1  After crossing paths at a party, a Cape Town t…
2  To protect his family from a powerful drug lor…
3  Feuds, flirtations and toilet talk go down amo…
4  In a city of coaching centers known to train I…
```

```
<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
```

[6]:
```
# Python program to convert .tsv file to .csv file
# importing pandas library

tsv_file_ratings='/Users/jaamiemaarshj/Desktop/  DAE Course Materials/
 ↪Computization and Visualisation/Mid term/title.ratings.tsv.gz'

tsv_file_basics='/Users/jaamiemaarshj/Desktop/  DAE Course Materials/
 ↪Computization and Visualisation/Mid term/title.basics.tsv.gz'

# reading given tsv file
csv_table_ratings=pd.read_table(tsv_file_ratings,sep='\t' , low_memory=False)
csv_title_basics=pd.read_table(tsv_file_basics,sep='\t' , low_memory=False)
```

```
# converting tsv file into csv
#csv_table_ratings.to_csv('title.ratings.csv',index=False)
#print("Successfully made ratings csv file")
#csv_title_basics.to_csv('title.basics.csv',index=False)
#print("Successfully made basics csv file")

df_title_ratings= pd.read_csv('title.ratings.csv', low_memory=False)
df_title_basics= pd.read_csv('title.basics.csv', low_memory=False).
 ↪rename(columns={'primaryTitle':'title', 'startYear':'release_year'})

display(df_title_ratings.head())

display(df_title_basics.head())
```

```
      tconst  averageRating  numVotes
0  tt0000001            5.7      1999
1  tt0000002            5.8       269
2  tt0000003            6.5      1888
3  tt0000004            5.5       178
4  tt0000005            6.2      2673
```

```
      tconst titleType                  title           originalTitle  \
0  tt0000001     short              Carmencita              Carmencita
1  tt0000002     short  Le clown et ses chiens  Le clown et ses chiens
2  tt0000003     short          Pauvre Pierrot          Pauvre Pierrot
3  tt0000004     short             Un bon bock             Un bon bock
4  tt0000005     short        Blacksmith Scene        Blacksmith Scene

  isAdult release_year endYear runtimeMinutes                    genres
0       0         1894      \N              1         Documentary,Short
1       0         1892      \N              5           Animation,Short
2       0         1892      \N              4  Animation,Comedy,Romance
3       0         1892      \N             12           Animation,Short
4       0         1893      \N              1              Comedy,Short
```

[7]:
```
# Merge title_df and rating_df
merged_ratings_basics = pd.merge(df_title_ratings, df_title_basics,␣
 ↪on='tconst', how='right')

# Replace '\\N' with NaN for better handling
merged_ratings_basics['release_year'].replace('\\N', pd.NA, inplace=True)
# Convert 'release_year' column to numeric and then to int64
merged_ratings_basics['release_year'] = pd.
 ↪to_numeric(merged_ratings_basics['release_year'], errors='coerce').
 ↪astype('Int64')
# Replace missing values with a default integer value, such as -1
merged_ratings_basics['release_year'].fillna(-1, inplace=True)
```

```
display(merged_ratings_basics.head())

merged_ratings_basics.info()
```

```
      tconst  averageRating  numVotes titleType                    title  \
0  tt0000001            5.7    1999.0     short                Carmencita
1  tt0000002            5.8     269.0     short  Le clown et ses chiens
2  tt0000003            6.5    1888.0     short           Pauvre Pierrot
3  tt0000004            5.5     178.0     short              Un bon bock
4  tt0000005            6.2    2673.0     short         Blacksmith Scene

            originalTitle isAdult  release_year endYear runtimeMinutes  \
0                Carmencita       0          1894      \N              1
1  Le clown et ses chiens       0          1892      \N              5
2           Pauvre Pierrot       0          1892      \N              4
3              Un bon bock       0          1892      \N             12
4         Blacksmith Scene       0          1893      \N              1

                     genres
0         Documentary,Short
1           Animation,Short
2  Animation,Comedy,Romance
3           Animation,Short
4              Comedy,Short
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10220465 entries, 0 to 10220464
Data columns (total 11 columns):
 #   Column          Dtype
---  ------          -----
 0   tconst          object
 1   averageRating   float64
 2   numVotes        float64
 3   titleType       object
 4   title           object
 5   originalTitle   object
 6   isAdult         object
 7   release_year    Int64
 8   endYear         object
 9   runtimeMinutes  object
 10  genres          object
dtypes: Int64(1), float64(2), object(8)
memory usage: 945.5+ MB
```

[9]:
```
#Both the netflix dataset and the combined title-ratings dataset is being␣
↪merged based on the below common fields
```

```
merged_netflix = pd.merge(df_netflix , merged_ratings_basics, on=['title' ,␣
 ↪'release_year'] , how='left')
display(merged_netflix.head())
merged_netflix.info()

print("End of data cleaning and processing ")
```

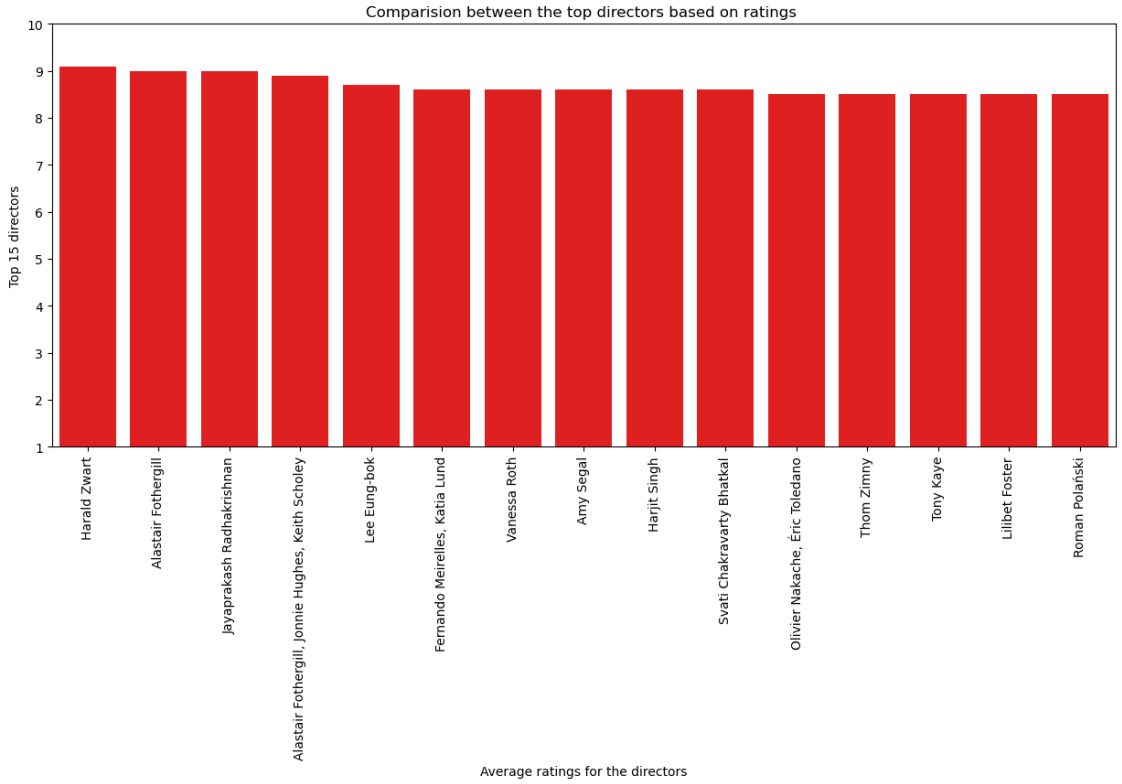
```
  show_id     type                  title         director  \
0      s1    Movie   Dick Johnson Is Dead  Kirsten Johnson
1      s2  TV Show           Blood & Water              NaN
2      s3  TV Show               Ganglands  Julien Leclercq
3      s4  TV Show  Jailbirds New Orleans              NaN
4      s5  TV Show            Kota Factory              NaN

                                                cast        country  \
0                                                NaN  United States
1  Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban…   South Africa
2  Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi…            NaN
3                                                NaN            NaN
4  Mayur More, Jitendra Kumar, Ranjan Raj, Alam K…          India

          date_added  release_year rating   duration  …  \
0  September 25, 2021          2020  PG-13     90 min  …
1  September 24, 2021          2021  TV-MA  2 Seasons  …
2  September 24, 2021          2021  TV-MA   1 Season  …
3  September 24, 2021          2021  TV-MA   1 Season  …
4  September 24, 2021          2021  TV-MA  2 Seasons  …

                                        description      tconst  \
0  As her father nears the end of his life, filmm…  tt11394180
1  After crossing paths at a party, a Cape Town t…  tt14810192
2  To protect his family from a powerful drug lor…  tt13278100
3  Feuds, flirtations and toilet talk go down amo…  tt15320436
4  In a city of coaching centers known to train I…         NaN

  averageRating  numVotes  titleType          originalTitle isAdult endYear  \
0           7.4    7069.0      movie   Dick Johnson Is Dead       0      \N
1           NaN       NaN      short          Blood & Water       0      \N
2           7.2    4229.0   tvSeries              Braqueurs       0      \N
3           6.6     278.0   tvSeries  Jailbirds New Orleans       0      \N
4           NaN       NaN        NaN                    NaN     NaN     NaN

  runtimeMinutes                      genres
0             89  Biography,Documentary,Drama
1             \N                 Drama,Short
2             44           Action,Crime,Drama
3             \N       Documentary,Reality-TV
```

```
4            NaN                          NaN

[5 rows x 21 columns]
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 12782 entries, 0 to 12781
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   show_id         12782 non-null  object
 1   type            12782 non-null  object
 2   title           12782 non-null  object
 3   director        9329 non-null   object
 4   cast            11783 non-null  object
 5   country         11709 non-null  object
 6   date_added      12769 non-null  object
 7   release_year    12782 non-null  int64
 8   rating          12778 non-null  object
 9   duration        12779 non-null  object
 10  listed_in       12782 non-null  object
 11  description     12782 non-null  object
 12  tconst          10113 non-null  object
 13  averageRating   6911 non-null   float64
 14  numVotes        6911 non-null   float64
 15  titleType       10113 non-null  object
 16  originalTitle   10113 non-null  object
 17  isAdult         10113 non-null  object
 18  endYear         10113 non-null  object
 19  runtimeMinutes  10113 non-null  object
 20  genres          10113 non-null  object
dtypes: float64(2), int64(1), object(18)
memory usage: 2.1+ MB
End of data cleaning and processing
```

[24]:
```
#Question 1:
#Create a bar chart visualization to emphasize the influence of directors on␣
 ↪Netflix based on
#their highest IMDb ratings

#Requirement: To create a bar chart between the directors and ratings

# step1: cleaning the rows having zeros/missing values or information
cleaned_netflix_data = merged_netflix.dropna(subset=['averageRating',␣
 ↪'director'])

# step2: to find the mean value of the ratings for the required directors
directors_vs_ratings = cleaned_netflix_data.
 ↪groupby('director')['averageRating'].mean().reset_index()
```

```
# step3: sorting the directors starting from the highest order by their␣
 ↪corresponding ratings
directors_vs_ratings = directors_vs_ratings.sort_values(by='averageRating',␣
 ↪ascending=False)

# step4: Selecting only the top 15 directors for better readability
highly_rated_directors = directors_vs_ratings.head(15)

# step5: plotting a bar chart showing the directors influence over the others.
plt.figure(figsize=(15, 6))
#the bar plot being displayed is by using seaborn
sns.barplot(x=highly_rated_directors['director'],␣
 ↪y=highly_rated_directors['averageRating'] , color='red')
#rotates the label values by 90 degree for better readability
plt.xticks(rotation=90)
#setting up the axis
plt.ylim( 1, 10, 1)
print(' Bar plot for the comparison between the directors and the ratings ')
plt.title('Comparision between the top directors based on ratings ')
plt.xlabel('Average ratings for the directors')
plt.ylabel('Top 15 directors')

plt.show()
```

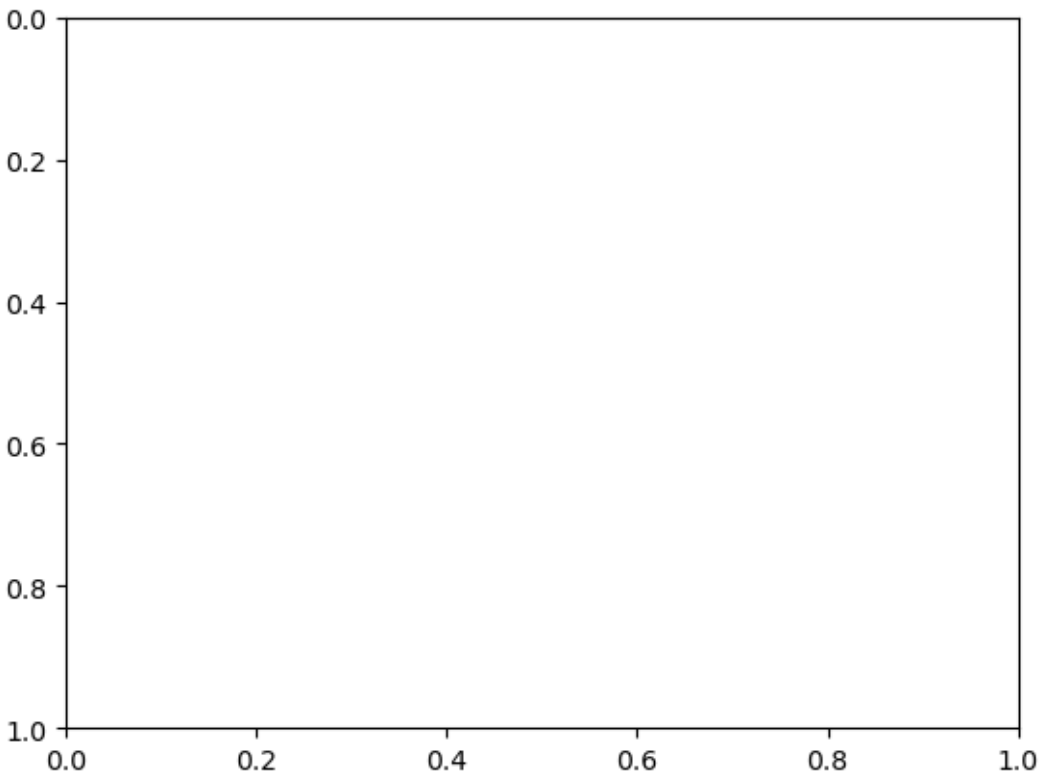


Insights:

From the above visualizaton it can be found that the top directors garner lot more viewership and intrest as inspite of taking into account the top 15 directors, everyone has an average rating of 8.75 out of 10 and this means that they are able to make the audiences tuned to their content and thereby increasing the viewship and the hours spent on watching Netflix

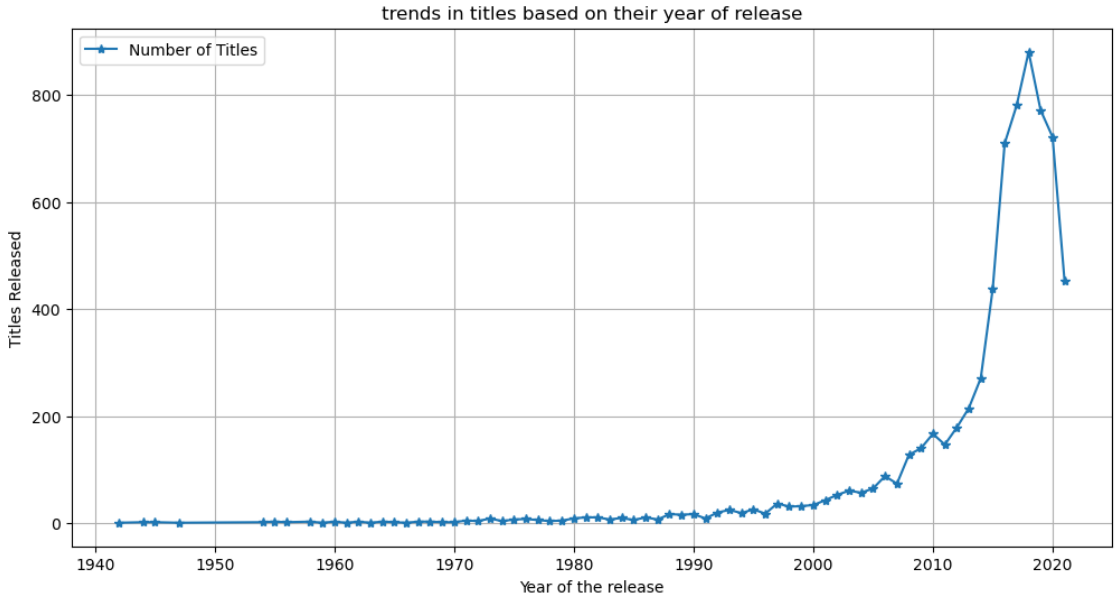
[47]:
```
#Question 2:

#Requirement: To create a grouped bar chart for top 15 genres for movies & TV␣
 ↪Shows

#step1: picking out specific information for movies/tv shows
#Filtering records based on Movie type
picking_movie_records = merged_netflix[merged_netflix['type'] == 'Movie']
#filtering based on TVshow
picking_TVshow_records = merged_netflix[merged_netflix['type'] == 'TV Show']

# Step2: bringing out the top 15 Movies genre
top_genre_movies = picking_movie_records['listed_in'].str.split(', ').explode().
 ↪value_counts().head(15)
# bringing out the top 15 Tv show genre
```

```
top_genres_TVshows = picking_TVshow_records['listed_in'].str.split(', ').
 ↪explode().value_counts().head(15)

# Step3: Creating a grouped barchart
grouped_bar_data = pd.DataFrame({
    'Genre': top_genres_TVshows.index,
    'Movies': top_genre_movies.values,
    'TV Shows': top_genre_movies.values
})

fig = px.bar(grouped_bar_data, x=['Movies', 'TV Shows'], y='Genre',
             title="List of highly watched TV shows and movies",
             labels={'value': 'Count', 'Genre': 'Genre'},
             width=800, height=500)

plt.gca().invert_yaxis()

fig.update_layout(barmode='group')
fig.update_xaxes(title_text='Genre')
fig.update_yaxes(title_text='Frequency/number')

fig.show()
```



Insights: from the above chart that the International Movies and TV shows are the hottest property in town since they are the most viewed once which are closely followed by dramas. So, we can infer that there are a lot of youngsters who are investing their times in watching netflix.

[48]:
```
#Question 3:

# requirement: to identify the trends in titles based on the year of release.

# Step1: cleaning out the zero data
cleaned_netflix_title_data = merged_netflix.dropna(subset=['release_year',␣
 ↪'averageRating'])

# Step2: counting the number of titles released every year
counts_year_release = cleaned_netflix_title_data['release_year'].value_counts().
 ↪reset_index()
counts_year_release.columns = ['Year', 'Title Count']

# Step3: Sorting of the data in their highest order
counts_year_release = counts_year_release.sort_values(by='Year')

# Step4: creating the visualization for the trends
plt.figure(figsize=(12, 6))
plt.plot(counts_year_release['Year'], counts_year_release['Title Count'],␣
 ↪marker='*', linestyle='-')
plt.title('trends in titles based on their year of release')
plt.xlabel('Year of the release')
plt.ylabel('Titles Released')

plt.legend(['Number of Titles'])
plt.grid(True)
plt.show()
```



Insights: For the visualisation, I am able to infer that the number of titles getting released has increased significantly over the years which also gives us that this is due to the fact that of the demand and supply concept and the number of titles drop after 2020 is because of COVID 19, which took a significant hit in the movie making where only a handful of movies/series were up for release

[51]:
```
#Question 4:

#Create an interactive bar chart that displays the distribution of content by␣
 ↪Top 10 countries.
#Note: Use plotly for interactive bar chart

#requirement: distribution by top 10 countries


# step1: cleaning the rows having zeros/missing values or information
cleaned_netflix_data = merged_netflix.dropna(subset=['country', 'title'])

# step2: to find the mean value of the ratings for the required directors
country_data = cleaned_netflix_data.groupby('country')['title'].count().
 ↪reset_index()

# step3: sorting the directors starting from the highest order by their␣
 ↪corresponding ratings
country_data = country_data.sort_values(by='title', ascending=False)

# step4: Selecting only the top 15 directors for better readability
```

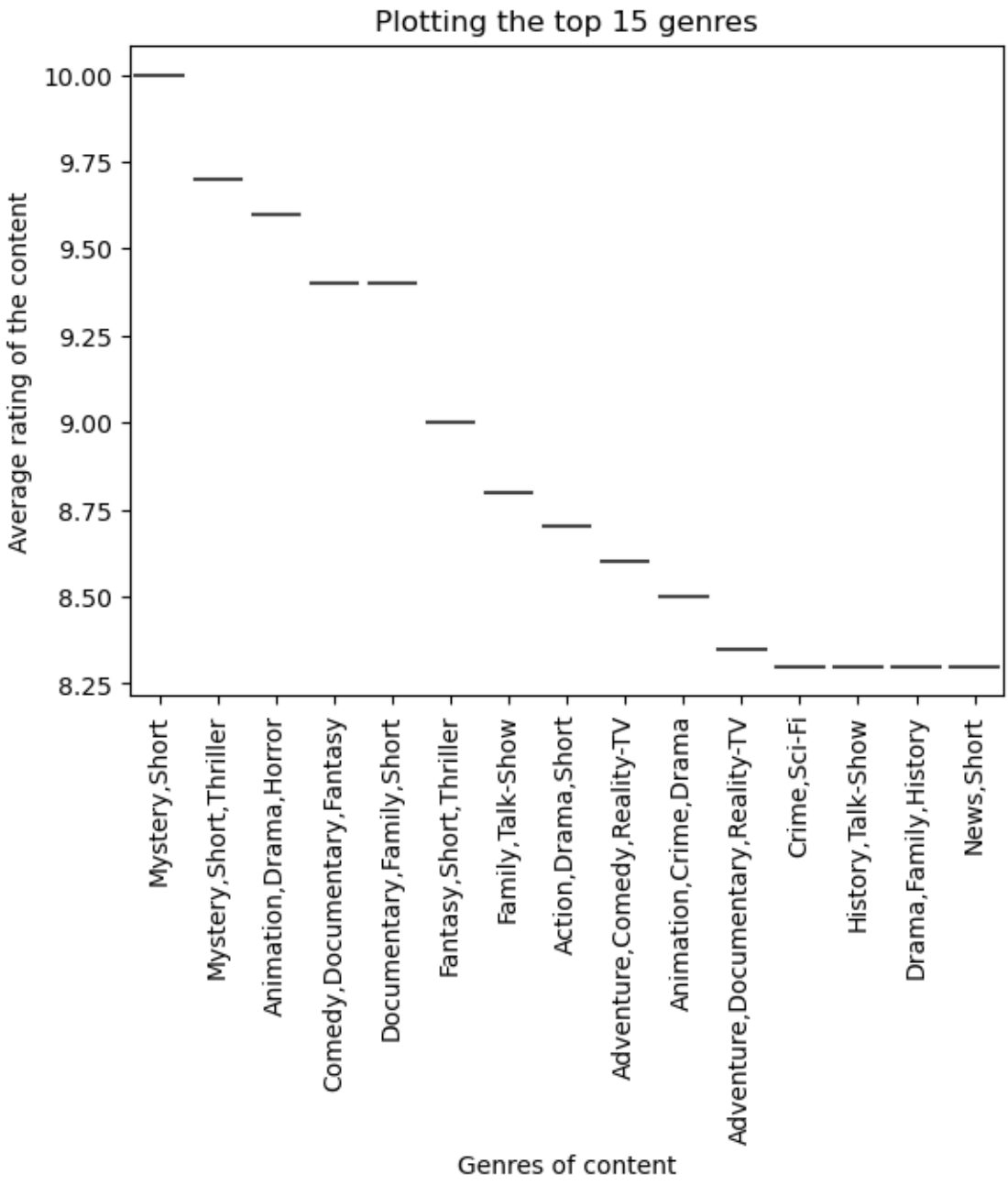
```
highly_country_data = country_data.head(10)

# Step5: bar chart for plotly
import plotly.express as px

# Sort the countries by average IMDb rating (optional)
#country_ratings = country_ratings.sort_values(by='averageRating',␣
 ↪ascending=False)

# Create a horizontal bar chart using Plotly
fig = px.bar(highly_country_data, x='title', y='country', orientation='h',
             labels={'title': 'No.of content', 'country': 'origin of the␣
 ↪content'},
             title='Top 10 Countries producing content on Netflix')
fig.update_layout(xaxis_title='No.of content', yaxis_title='origin of the␣
 ↪content')
fig.update_yaxes(categoryorder='total ascending')  # Invert the y-axis to␣
 ↪display the highest rating at the top

fig.show()
```

Insights: It can be found that the US stands way ahead interms of total content produced to around 4400 counts untill 2020 data stats. This large number of content could be due to the fact of the quality of content getting produced and its viewer ship which have a well eshtablished base. The countries following America are still finding their feet as they have cracked the Netflix concept post COVID era.

[60]:
```
#Question 5:

#requirement: To visualize top genres based on IMDB

# Step1: grouping based on genre
top_genres_ratings = merged_netflix.groupby('genres')['averageRating'].mean().
 ↪reset_index()

# Step2; Sorting the generes based on their highest rating
top_genres_ratings = top_genres_ratings.sort_values(by='averageRating',␣
 ↪ascending=False)

#Step3: Taking only top 15 genres
top15_genres_ratings = top_genres_ratings.head(15)

display(top15_genres_ratings)

sns.violinplot(data=top15_genres_ratings, x="genres", y="averageRating")

plt.title('Plotting the top 15 genres')
```

```
plt.xlabel('Genres of content')
plt.ylabel('Average rating of the content')

plt.xticks(rotation=90)

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

	genres	averageRating
597	Mystery,Short	10.00
598	Mystery,Short,Thriller	9.70
205	Animation,Drama,Horror	9.60
272	Comedy,Documentary,Fantasy	9.40
407	Documentary,Family,Short	9.40
547	Fantasy,Short,Thriller	9.00
528	Family,Talk-Show	8.80
68	Action,Drama,Short	8.70
139	Adventure,Comedy,Reality-TV	8.60
196	Animation,Crime,Drama	8.50
148	Adventure,Documentary,Reality-TV	8.35
390	Crime,Sci-Fi	8.30
561	History,Talk-Show	8.30
441	Drama,Family,History	8.30
603	News,Short	8.30

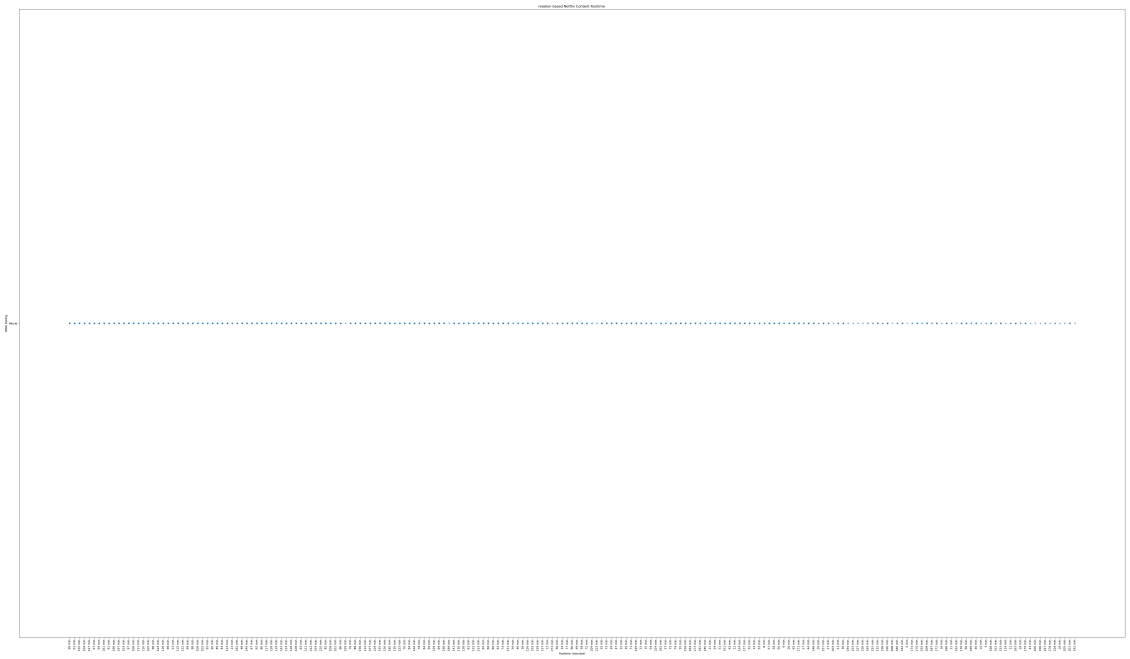


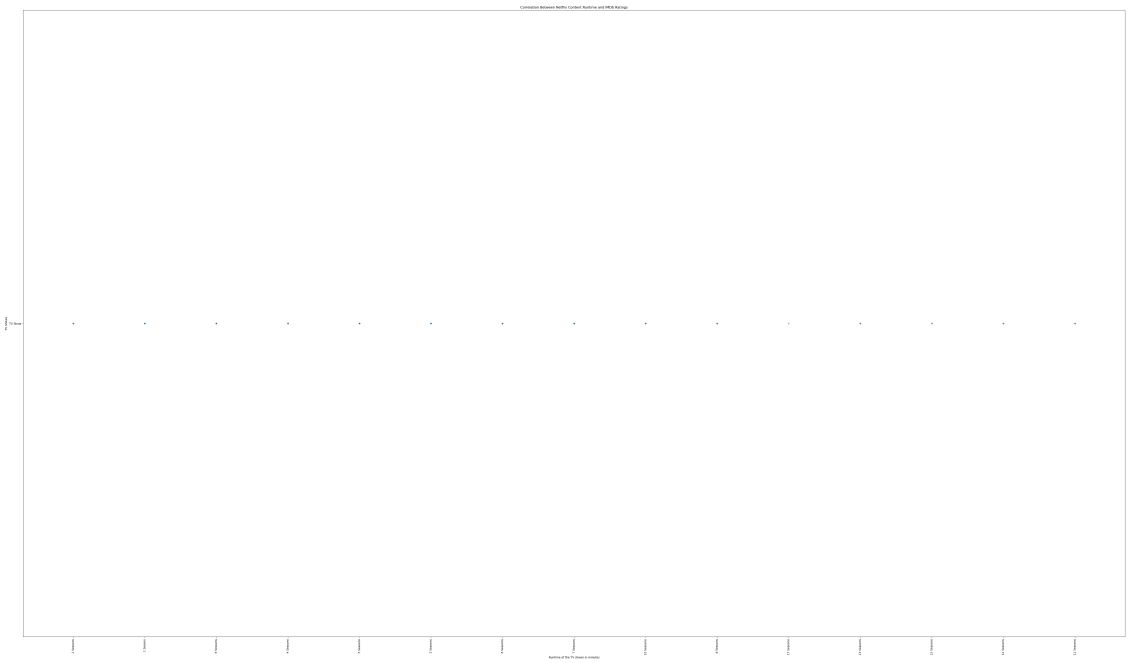
Insights: It can be observed that the mystery genre as a whole has garned the highest rating whether the content was short or long since the short news content couldnt garner much rating as they seem to be less engaging than the mystery one.

[66]:
```
#Question 6

#requirement: distribution between length of movies and shows
#
```

```
picking_movie_records = merged_netflix[merged_netflix['type'] == 'Movie']
#filtering based on TVshow
picking_TVshow_records = merged_netflix[merged_netflix['type'] == 'TV Show']

# plotting the scatter plot for movies
plt.figure(figsize=(70,40))
sns.scatterplot(x='duration', y='type', data=picking_movie_records, alpha=0.5)
plt.title('relation based Netflix Content Runtime ')
plt.xlabel('Runtime (minutes)')
plt.xticks(rotation=90)
plt.ylabel('IMDb Rating')

# plotting the scatter plot for Tv shows
plt.figure(figsize=(70,40))
sns.scatterplot(x='duration', y='type', data=picking_TVshow_records, alpha=0.5)
plt.title('Correlation Between Netflix Content Runtime and IMDb Ratings')
plt.xlabel('Runtime of the TV shows in minutes')
plt.xticks(rotation=90)
plt.ylabel('TV Shows')
```

[66]: Text(0, 0.5, 'TV Shows')





Insights: it can be found that across both the types, it can be concluded that the watch time doent matter if the content is good.

[68]:
```
#Question 7:
# requirement: movie duration over the years


# step1: deopping out rows with missing IMDb ratings, release years, and␣
 ↪runtime values
filtered_netflix = merged_netflix.dropna(subset=['averageRating',␣
 ↪'release_year', 'duration'])

# Step 2: Checking for unique values
Netflix_content_durations = filtered_netflix['duration'].unique()

# step3: Identifying the format of durations,
duration_format = 'min'
# step 4: Extract runtime as a new column
filtered_netflix['runtime_minutes'] = filtered_netflix['duration'].str.
 ↪extract(f'(\d+) {duration_format}').astype(float)

# step5: plotting the Scatter plot
fig1 = px.scatter(filtered_netflix, x='release_year', y='runtime_minutes',
                  labels={'release_year': 'Years', 'runtime_minutes': 'Content␣
 ↪runtime in minutes'},
```

```
                  title='Relationship Between Content Duration over the years␣
 ↪on Netflix')
fig1.update_traces(marker=dict(size=5, opacity=0.5))
```

Insights: The content duration across the years have reduced more since 2010 to 2020

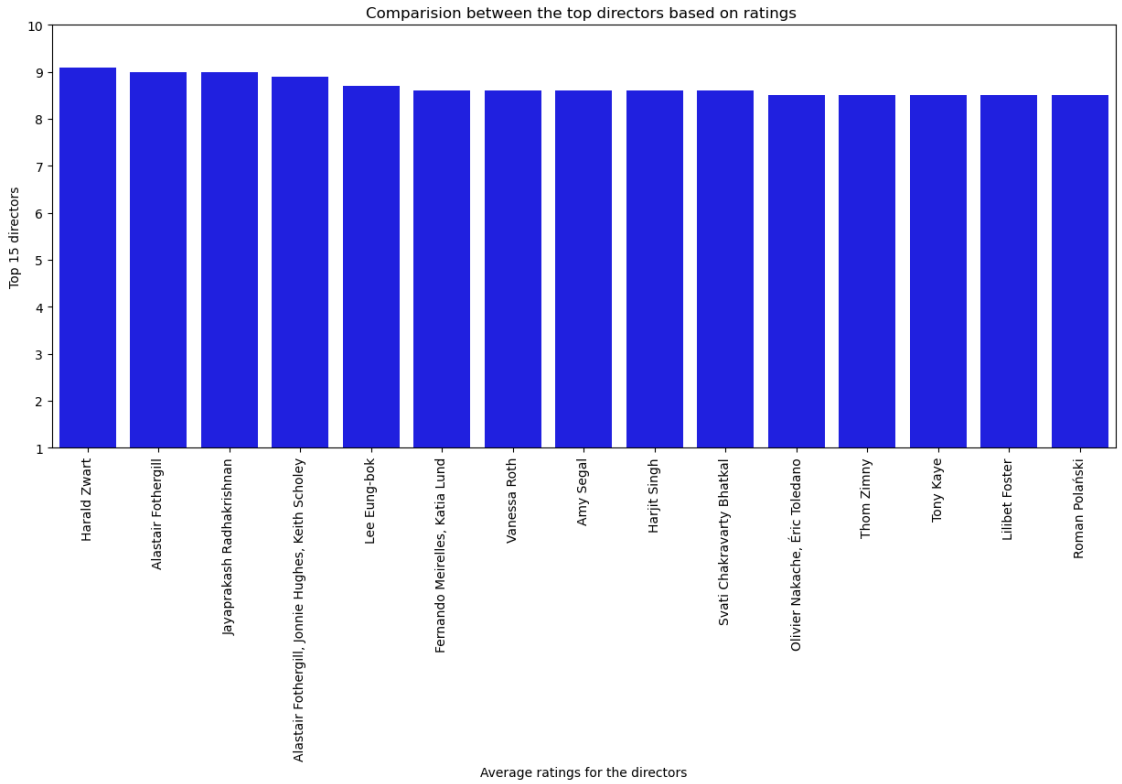
[83]:
```
#question 8:

#requirement:  distribution of IMDB ratings for Netflix titles grouped by␣
 ↪release year

df_rating = merged_netflix.groupby('release_year')['averageRating'].mean().
 ↪reset_index()

fig = px.line(df_rating, x='release_year', y='averageRating',
              title='Ratings Trends for Netflix Titles ',
              markers=True)

fig.update_xaxes(dtick=1, title_text='Release Year')
fig.update_yaxes(title_text='Average Rating for titles')

fig.show()
```

Insights: It is found the over the years the rating has been the highest in the year 1962 and has been constant throughout

[82]:
```
#Question 9:


#Requirement: To create a bar chart between the directors and greatest content

# step1: cleaning the rows having zeros/missing values or information
cleaned_netflix_data = merged_netflix.dropna(subset=['averageRating',␣
 ↪'director'])

# step2: to find the mean value of the ratings for the required directors
directors_vs_ratings = cleaned_netflix_data.
 ↪groupby('director')['averageRating'].mean().reset_index()

# step3: sorting the directors starting from the highest order by their␣
 ↪corresponding ratings
directors_vs_ratings = directors_vs_ratings.sort_values(by='averageRating',␣
 ↪ascending=False)

# step4: Selecting only the top 15 directors for better readability
highly_rated_directors = directors_vs_ratings.head(15)
```

```
# step5: plotting a bar chart showing the directors influence over the others.
plt.figure(figsize=(15, 6))
#the bar plot being displayed is by using seaborn
sns.barplot(x=highly_rated_directors['director'],␣
 ↪y=highly_rated_directors['averageRating'] , color='blue')
#rotates the label values by 90 degree for better readability
plt.xticks(rotation=90)
#setting up the axis
plt.ylim( 1, 10, 1)
print(' Bar plot for the comparison between the directors and the ratings ')
plt.title('Comparision between the top directors based on ratings ')
plt.xlabel('Average ratings for the directors')
plt.ylabel('Top 15 directors')

plt.show()
```

```
Bar plot for the comparison between the directors and the ratings
```



Insights: Harald Zwart has been the one who has done the greatest number of content.

[86]:
```
#Ques 10:
```

```
#requirement: netflix content over the years

cleaned_netflix_data['released year'] = pd.
 ↪to_datetime(cleaned_netflix_data['date_added']).dt.year
year_counts = cleaned_netflix_data['released year'].value_counts().reset_index()

year_counts.columns = ['year on netflix' , 'count of the titles']

fig = px.line(year_counts, x='year on netflix', y='count of the titles',␣
 ↪labels={'year on netflix': 'year on netflix',  'count of the titles': 'count␣
 ↪of the titles'},
              title='Trends in content Over Years on Netflix (Movies)')

fig.update_xaxes(title='trends across the years')
fig.update_yaxes(title='Average IMDb Rating')

fig.show()
```

Insights: the more number of titles have been in 2019 whereas its almost null untill 2015. post the COVID era, the count in the content has dropped significantly as there has no movies which has been shot.