IMDB MOVIE ANALYSIS WITH PYTHON

- Project Setup and Data Loading

- Loaded the dataset in the tabular form for better understanding

- Table includes columns of Movie Names, release date_x, score, genre, overview, crew names, orig_title, status, orig_language, Budget of the movie, Revenue of the movie, Country

- Importing libraries like pandas, numpy, seaborn and matplotlib for analysis

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

When it comes to Analysing data through data visualization in Python, three libraries stand out: Matplotlib, Pandas, and Seaborn. Each of these libraries has its own strengths and use cases.

Matplotlib is a fundamental library for creating static, animated, and interactive visualizations in Python. It provides a wide range of plotting functionalities and is highly customizable. Matplotlib is ideal for creating basic plots and customizing them to meet specific requirements.

Pandas, a popular data manipulation library, also offers basic plotting capabilities through its integration with Matplotlib. Pandas simplifies the process of creating visualizations from DataFrames and Series, making it convenient for quick exploratory data analysis.

Seaborn is built on top of Matplotlib and provides a higher-level interface for creating informative and attractive statistical graphics. It simplifies the process of creating complex visualizations by providing built-in themes and statistical functionalities.

```
df = pd.read_csv("imdb_movies.csv")
df
```

```
                                        names       date_x  score  \
0                                   Creed III  03/02/2023    73.0
1                     Avatar: The Way of Water  12/15/2022    78.0
2                  The Super Mario Bros. Movie  04/05/2023    76.0
3                                     Mummies  01/05/2023    70.0
4                                   Supercell  03/17/2023    61.0
...                                       ...         ...    ...
10173                     20th Century Women  12/28/2016    73.0
10174  Delta Force 2: The Colombian Connection  08/24/1990    54.0
10175                       The Russia House  12/21/1990    61.0
10176          Darkman II: The Return of Durant  07/11/1995    55.0
10177       The Swan Princess: A Royal Wedding  07/20/2020    70.0

                                              genre  \
0                                     Drama, Action
```

```
1                      Science Fiction, Adventure, Action
2          Animation, Adventure, Family, Fantasy, Comedy
3          Animation, Comedy, Family, Adventure, Fantasy
4                                                  Action
...                                                   ...
10173                                              Drama
10174                                             Action
10175                          Drama, Thriller, Romance
10176  Action, Adventure, Science Fiction, Thriller, ...
10177                        Animation, Family, Fantasy

                                                 overview  \
0      After dominating the boxing world, Adonis Cree...
1      Set more than a decade after the events of the...
2      While working underground to fix a water main,...
3      Through a series of unfortunate events, three ...
4      Good-hearted teenager William always lived in ...
...                                                  ...
10173  In 1979 Santa Barbara, California, Dorothea Fi...
10174  When DEA agents are taken captive by a ruthles...
10175  Barley Scott Blair, a Lisbon-based editor of R...
10176  Darkman and Durant return and they hate each o...
10177  Princess Odette and Prince Derek are going to ...

                                                    crew  \
0      Michael B. Jordan, Adonis Creed, Tessa Thompso...
1      Sam Worthington, Jake Sully, Zoe Saldaña, Neyt...
2      Chris Pratt, Mario (voice), Anya Taylor-Joy, P...
3      Óscar Barberán, Thut (voice), Ana Esther Albor...
4      Skeet Ulrich, Roy Cameron, Anne Heche, Dr Quin...
...                                                  ...
10173  Annette Bening, Dorothea Fields, Lucas Jade Zu...
10174  Chuck Norris, Col. Scott McCoy, Billy Drago, R...
10175  Sean Connery, Bartholomew 'Barley' Scott Blair...
10176  Larry Drake, Robert G. Durant, Arnold Vosloo, ...
10177  Nina Herzog, Princess Odette (voice), Yuri Low...

                                      orig_title     status  \
0                                      Creed III   Released
1                        Avatar: The Way of Water   Released
2                    The Super Mario Bros. Movie   Released
3                                         Momias   Released
4                                      Supercell   Released
...                                          ...        ...
10173                        20th Century Women   Released
10174  Delta Force 2: The Colombian Connection   Released
10175                          The Russia House   Released
10176          Darkman II: The Return of Durant   Released
10177       The Swan Princess: A Royal Wedding   Released
```

```
                  orig_lang     budget_x       revenue country
0                   English   75000000.0  2.716167e+08      AU
1                   English  460000000.0  2.316795e+09      AU
2                   English  100000000.0  7.244590e+08      AU
3       Spanish, Castilian   12300000.0  3.420000e+07      AU
4                   English   77000000.0  3.409420e+08      US
...                     ...          ...           ...     ...
10173               English    7000000.0  9.353729e+06      US
10174               English    9145817.8  6.698361e+06      US
10175               English   21800000.0  2.299799e+07      US
10176               English  116000000.0  4.756613e+08      US
10177               English   92400000.0  5.394018e+08      GB

[10178 rows x 12 columns]
```

Data Overview and Basic Exploration, Identifying Missing Values

```
#With df.info we will get the basic idea of all the data entries, columns, null-values, data types and all the potential issues with the data
```

Using .info() to understand the data types and missing values

```
df.info()               #Detailed Info of the data

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10178 entries, 0 to 10177
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   names       10178 non-null  object 
 1   date_x      10178 non-null  object 
 2   score       10178 non-null  float64
 3   genre       10093 non-null  object 
 4   overview    10178 non-null  object 
 5   crew        10122 non-null  object 
 6   orig_title  10178 non-null  object 
 7   status      10178 non-null  object 
 8   orig_lang   10178 non-null  object 
 9   budget_x    10178 non-null  float64
 10  revenue     10178 non-null  float64
 11  country     10178 non-null  object 
```

```
dtypes: float64(3), object(9)
memory usage: 954.3+ KB
```

Describe the main characteristics of each column using .describe()

```
df.describe()
```

```
               score      budget_x       revenue
count  10178.000000  1.017800e+04  1.017800e+04
mean      63.497052  6.488238e+07  2.531401e+08
std       13.537012  5.707565e+07  2.777880e+08
min        0.000000  1.000000e+00  0.000000e+00
25%       59.000000  1.500000e+07  2.858898e+07
50%       65.000000  5.000000e+07  1.529349e+08
75%       71.000000  1.050000e+08  4.178021e+08
max      100.000000  4.600000e+08  2.923706e+09
```

Handling Null Values, Data Cleaning, Converting Data Types

```
df["date_x"] = pd.to_datetime(df["date_x"])
#Converts the date from object to time


df.isnull().sum()
#Will give the sum of all null values
```

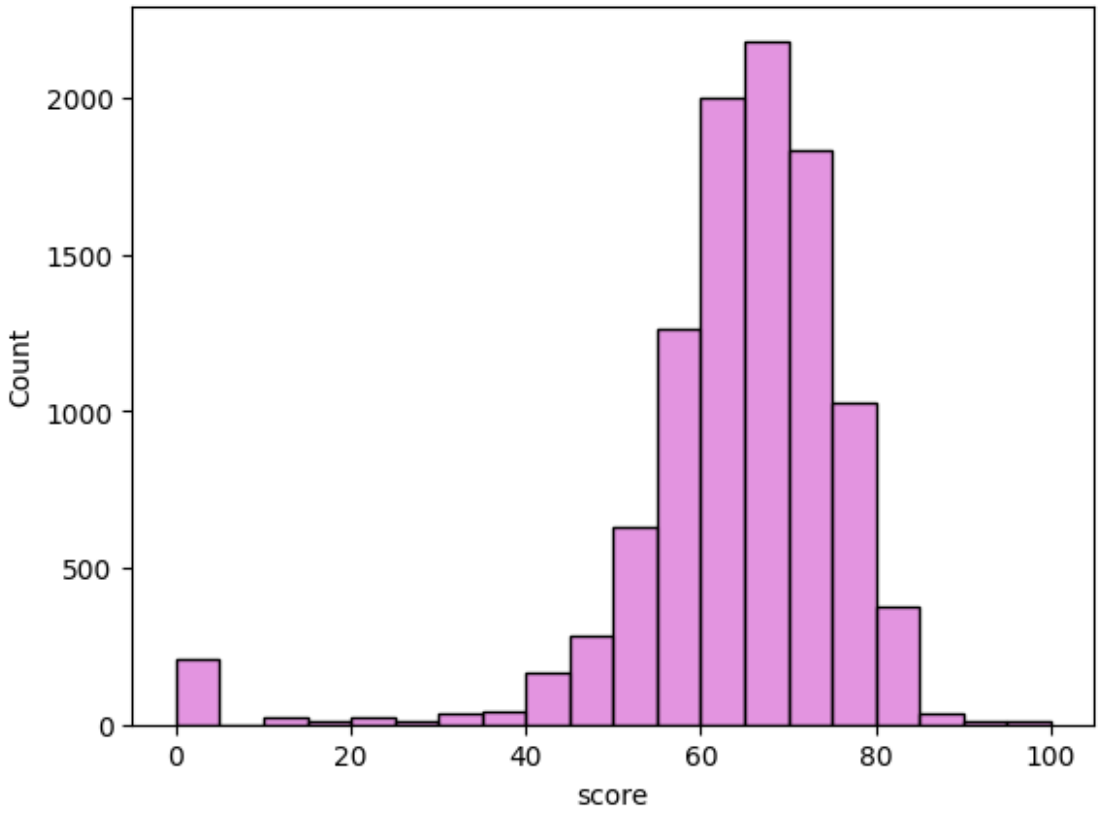
```
names          0
date_x         0
score          0
genre         85
overview       0
crew          56
orig_title     0
status         0
orig_lang      0
budget_x       0
revenue        0
country        0
dtype: int64
```

Filling Null Values

```
df['genre'] = df["genre"].fillna("unavailable")     #Filling null
values


df["crew"] = df["crew"].fillna("unavailable")      #Filling null
values



#Importing the necessary info -- Done
#Checked for info             -- Done
#Inspected the info           -- Done
#Data Cleaning                -- Done
#Replacing the null values    -- Done
```

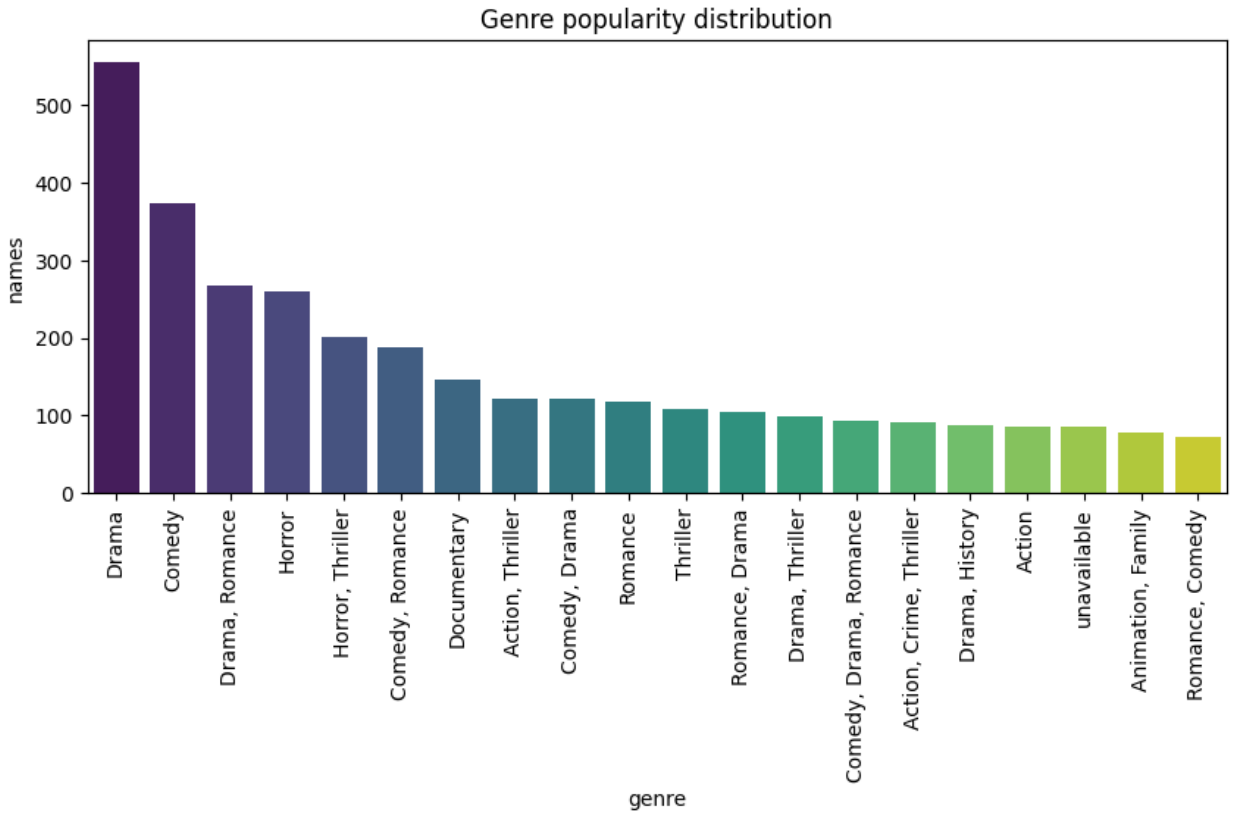
Distribution based on Movie Score on a Histogram

```
sns.histplot(x = "score", data = df, bins = 20, color="orchid")
plt.show()
```



Showing Genre Popularity distribution - Common Genres in Dataset

```
gb = df.groupby("genre").agg({"names":"count"})
gb = gb.sort_values(by = "names",ascending = False)
gb = gb.head(20)
plt.figure(figsize =(10,4))
sns.barplot(x = gb.index, y = gb["names"], data = gb, hue = gb.index,
palette = "viridis")
plt.title("Genre popularity distribution")
plt.xticks(rotation = 90)
plt.show()
```



Analysing Genre Popularity Distribution

KEY INSIGHTS:

Drama Dominates: *The genre "Drama" is the most popular, with a significantly higher count compared to other genres. This suggests that viewers have a strong preference for dramatic storytelling.*

Comedy and Romance Follow: *"Comedy" and "Romance" genres hold the second and third positions, indicating a considerable audience interest in light-hearted and romantic content.*

Genre Combinations: *Genre combinations like "Drama, Romance" and "Comedy, Romance" are quite popular, suggesting that viewers often enjoy movies that blend genres.*

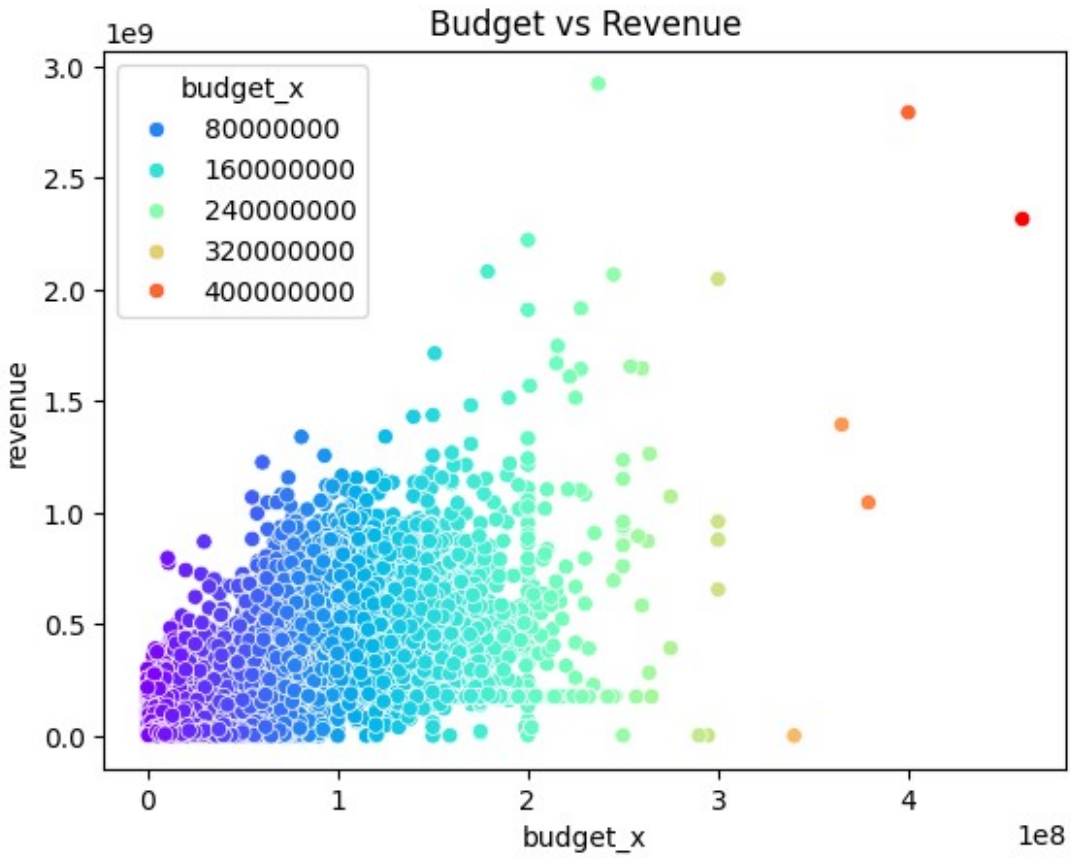
Thriller and Action: *Genres like "Thriller" and "Action" are also well-represented, indicating a significant audience for suspenseful and high-octane content.*

Niche Genres: *Genres like "Documentary," "Animation," and "Family" have lower counts, suggesting a smaller but dedicated audience for these specific types of films.*

Overall: *The chart highlights the diverse preferences of viewers, with a strong preference for drama, comedy, and romance. It also shows a notable interest in genre combinations and action-packed content.*

Scatter Plot for Budget vs Revenue

```
plt.title("Budget vs Revenue")
sns.scatterplot(x = "budget_x", y = "revenue", data = df,
hue=df.budget_x, palette="rainbow")
plt.show()
```



Insights from the Scatter plot above

Positive Correlation:

The chart shows a strong positive correlation between budget and revenue. As the budget increases, the revenue tends to increase significantly, though with some variance.

Diminishing Returns at Higher Budgets:

For budgets over 3×10^8 (300M), the number of movies decreases, and only a few movies achieve exceptionally high revenue (>2.5×10^9 (2.5B)). This suggests high-budget movies can generate blockbuster revenues but remain a high-risk investment due to their smaller volume.

Clusters by Budget Bracket:

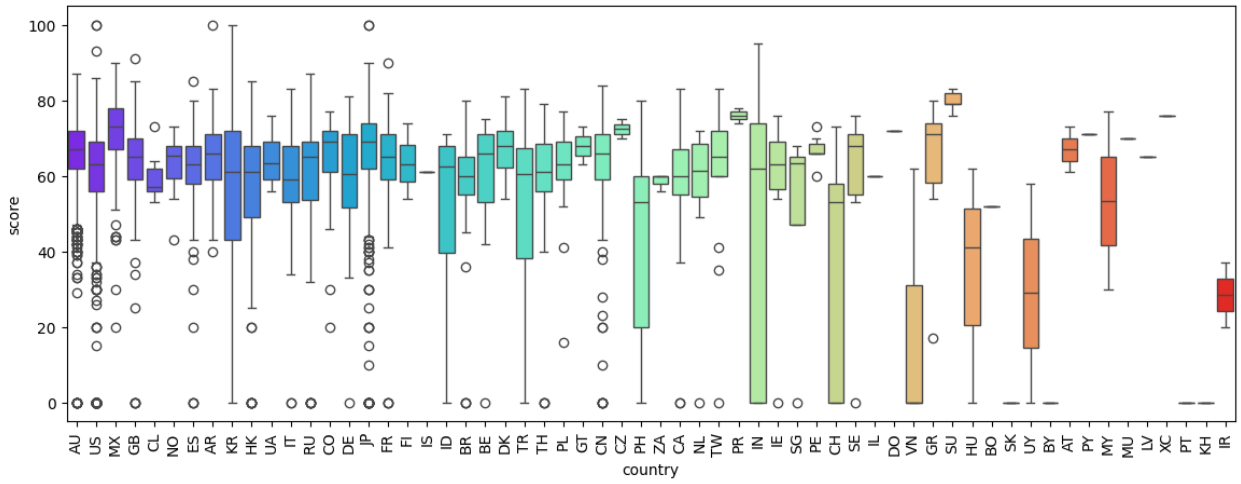
The color-coded clusters reveal distinct budget categories. The majority of movies fall within the lower budget range, highlighting that most productions operate with limited budgets, achieving moderate revenue results.

Conclusion:

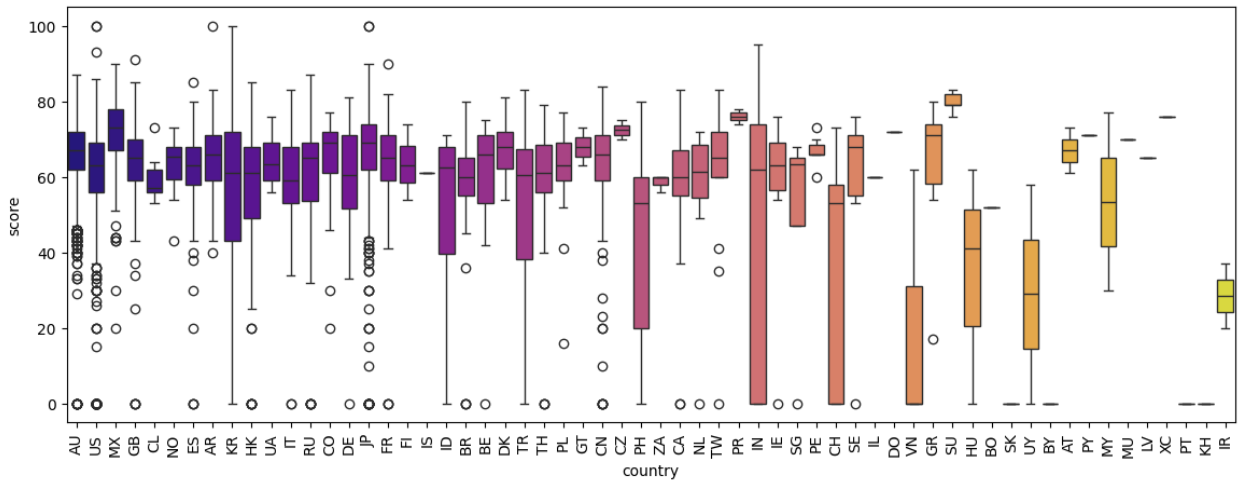
This analysis highlights that while higher budgets often lead to higher revenues, there's no guarantee of success, especially beyond certain budget thresholds. Most movies achieve moderate financial success within standard budget ranges.

Boxplot to visualize the differences in ratings across different countries

```
#Score vary by country (Box plot)
plt.figure(figsize = (14,5))
sns.boxplot(x = "country", y = "score", data = df, hue = df.country,
palette="rainbow")
plt.xticks(rotation = 90)
plt.show()
```

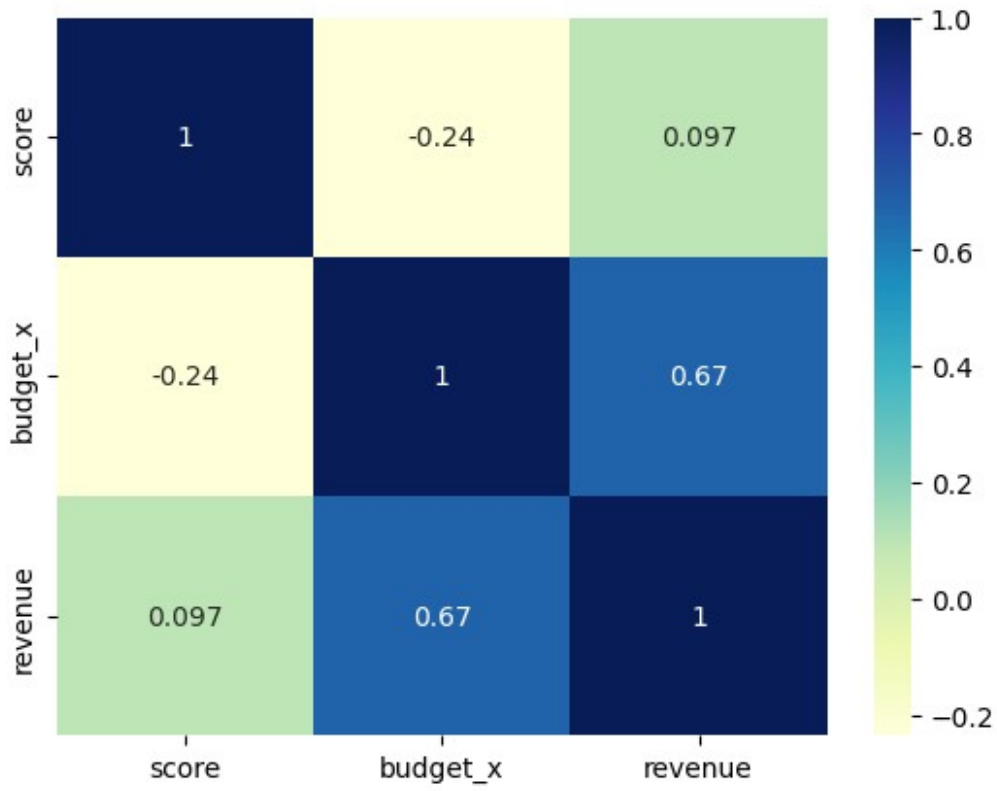



```
plt.figure(figsize=(14, 5))
sns.boxplot(x = "country", y ="score", data = df, hue = df.country,
palette="plasma")     # Add a color palette here
plt.xticks(rotation=90)
plt.show()
```



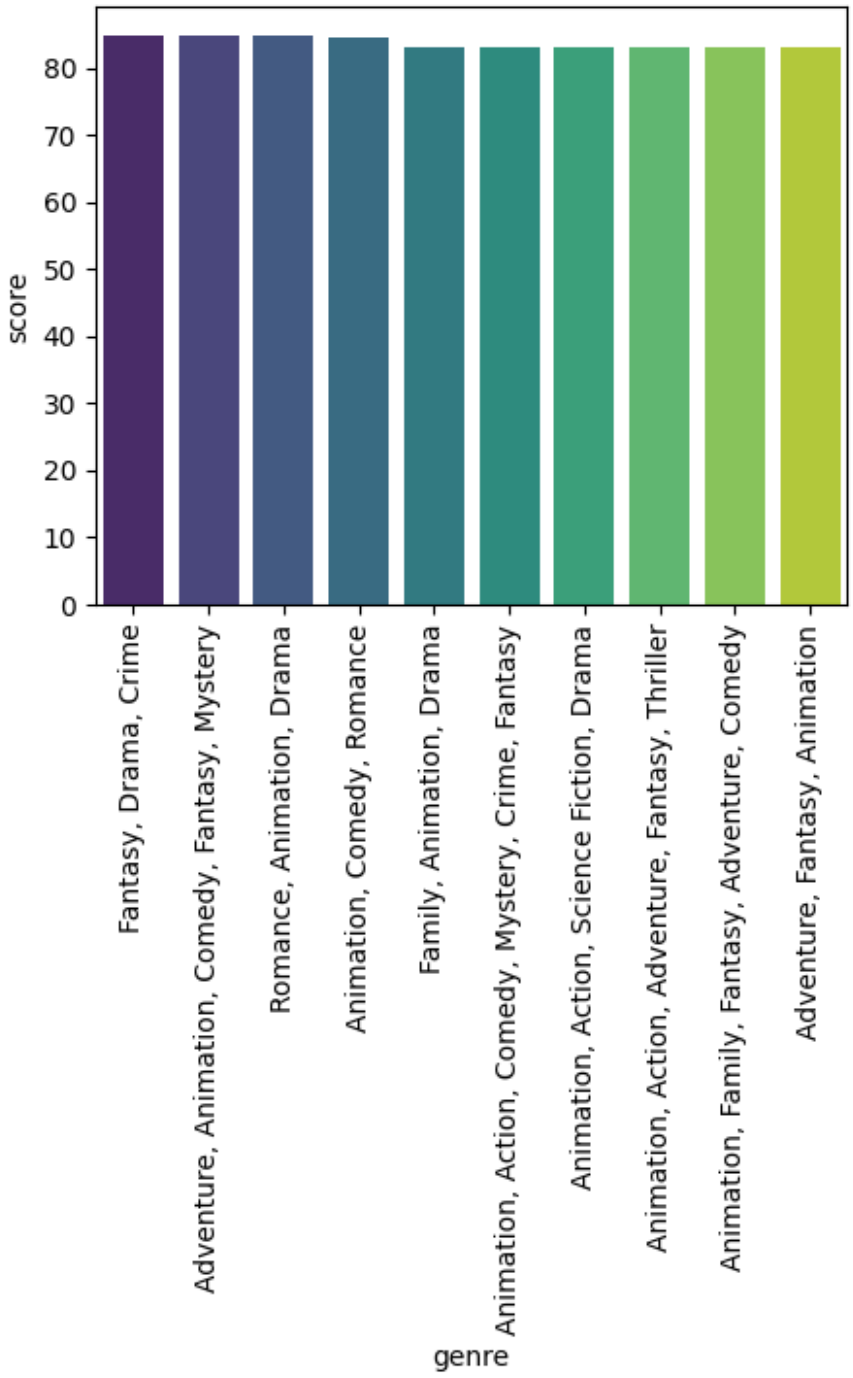
Heatmap representing Correlation between Score, Budget and Revenue

```
df1 = df[["score", "budget_x", "revenue"]]
c = df1.corr()
sns.heatmap(c, annot = True, cmap= "YlGnBu")                    #To get
the annotataions
plt.show()
```



Genre with Highest Average Rating. Calculating the Average Rating for each Genre

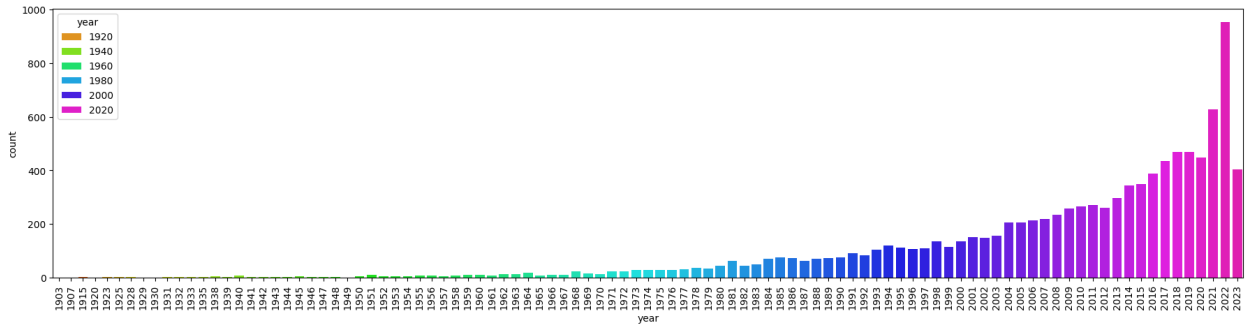
```
gb = df.groupby("genre").agg({"score":"mean"})
gb = gb.sort_values(by = "score",ascending = False)
gb = gb.head(10)
plt.figure(figsize = (5,4))
sns.barplot(x = gb.index, y = gb["score"], data = gb, hue = gb.index,
palette = "viridis")
plt.xticks(rotation = 90)
plt.show()
```




```
df ["year"] = df["date_x"].dt.strftime("%Y")
```

Countplot to show number of movies released per year

```
df ["year"] = df["date_x"].dt.strftime("%Y")
df["year"] = df['year'].astype('int')
plt.figure(figsize = (22,5))
sns.countplot(x = "year", data = df, hue = df.year, palette =
"gist_rainbow")
plt.xticks(rotation = 90, fontsize = 10)
plt.show()
```



Analysis :

The number of movies released per year has seen a significant increase over time, with a particularly sharp rise in recent decades.

Specific Decades: *The 1920s and 1930s had relatively fewer movie releases compared to later decades. The 1940s experienced a slight dip in movie production. From the 1950s onwards, there was a steady increase in movie releases, with the 2010s marking the decade with the highest number of movies. 2022 has seen the highest number of movie releases being the golden year for film fraternity*

```
df["year"] = df['year'].astype('int')
```

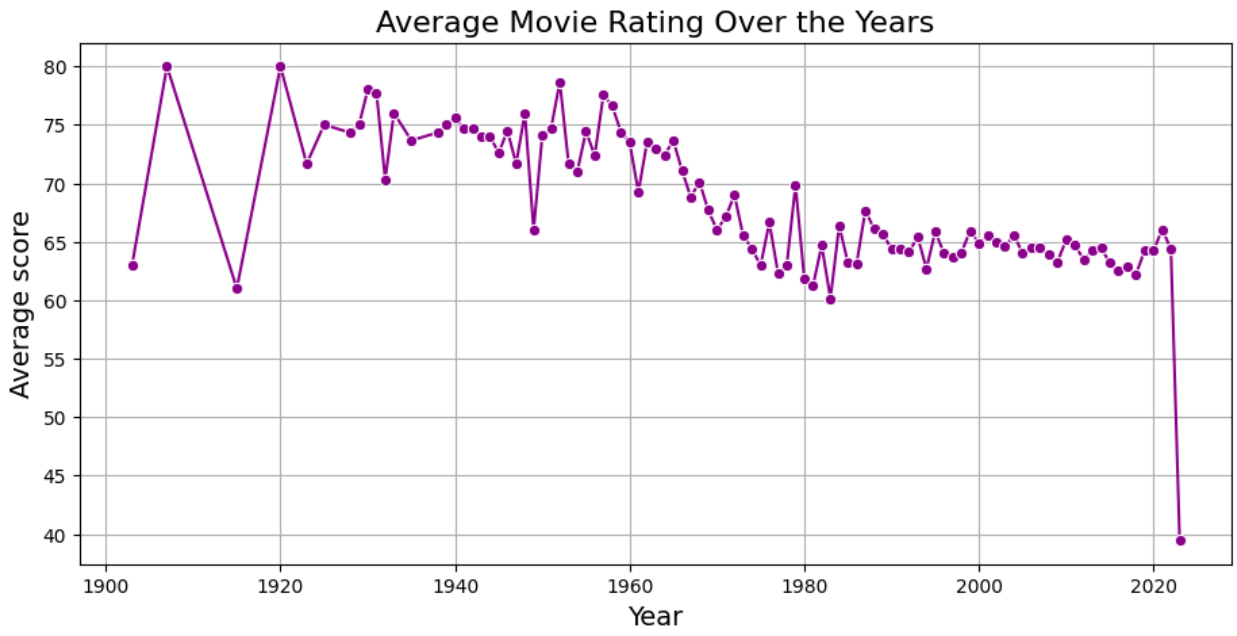
Popularity of genres over Time. Plotting the number of movies released per genre each year.

```
gb = df.groupby(["year", "genre"]).agg({"date_x":"count"})
gb
```

```
                                      date_x
year genre
1903 Drama, History                        1
1907 Adventure, Science Fiction            1
1915 Drama, History, War                   2
1920 Drama, Horror, Thriller, Crime        1
1923 Comedy, Romance, Thriller             1
...                                      ...
2023 War, Drama, History                   1
     War, History, Drama                   1
     Western                               1
     Western, Action                       1
     unavailable                          24

[6438 rows x 1 columns]
```

Plotting Average Movie Rating Over the Years

```
# Load your data
# Ensure the 'year' and 'rating' columns exist in your dataset
# Group data by year and calculate the average rating

avg_rating_per_year = df.groupby('year')['score'].mean().reset_index()

# Plot using Seaborn or Matplotlib

plt.figure(figsize=(11, 5))
sns.lineplot(data=avg_rating_per_year, x='year', y='score',
marker='o', color='darkmagenta')  # Add marker for visibility
plt.title('Average Movie Rating Over the Years', fontsize=16)
plt.xlabel('Year', fontsize=14)
plt.ylabel('Average score', fontsize=14)
plt.grid(True)            # Add grid for better readability
plt.show()
```



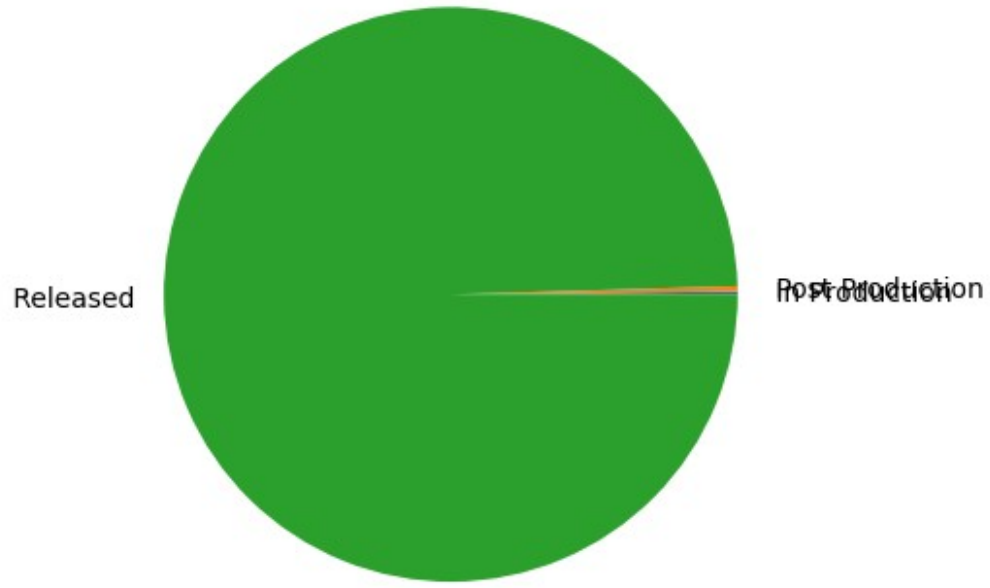
Analysing Average Movie Ratings over the Years

Early Fluctuations: *Ratings before 1950 display large variability, suggesting inconsistent quality, smaller data samples, or varying criteria for rating movies during the earlier decades. Gradual Decline: Over time, the average ratings have trended downwards. This could indicate stricter audience expectations, shifts in rating standards, or changing perceptions of quality in cinema.*

Sharp Drop (2020): *The dramatic fall in the 2020s is an outlier and might need further investigation. Potential reasons could include: Disruption caused by global events like the COVID-19 pandemic. A small dataset for recent years leading to skewed average*

Presenting the Status of Movies through a Piechart

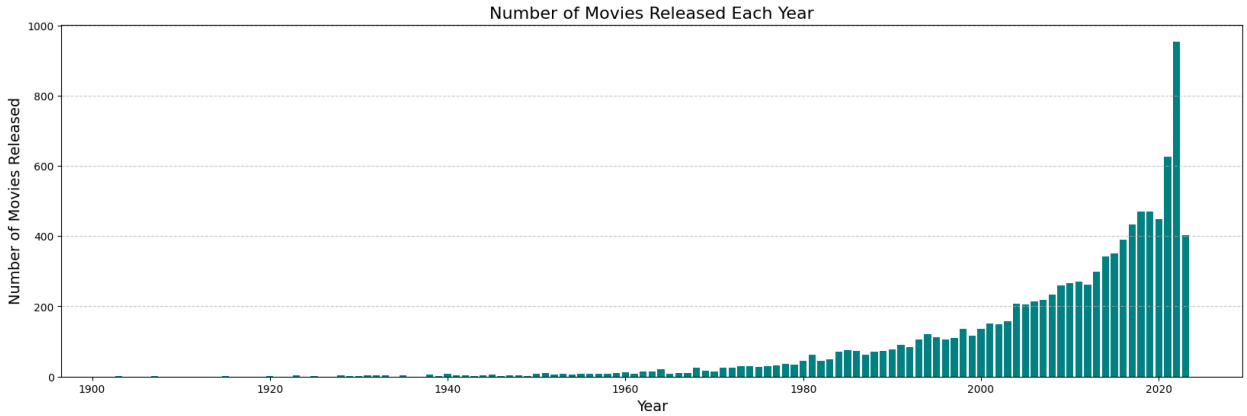
```
gb = df.groupby("status").agg({"score":"count"})
plt.pie(gb["score"],labels = gb.index)
plt.show()
```



Plotting the number of movies released each year(Years with high and low number of movie releases)

```
#Q7.ii

# Group by year and count the number of movies released
movies_per_year = df['year'].value_counts().sort_index()

# Find the years with the highest and lowest number of releases
max_releases_year = movies_per_year.idxmax()
min_releases_year = movies_per_year.idxmin()
max_releases = movies_per_year.max()
min_releases = movies_per_year.min()

print(f"The year with the highest number of movie releases:
{max_releases_year} ({max_releases} movies)")
print(f"The year with the lowest number of movie releases:
{min_releases_year} ({min_releases} movies)")

# Plot the number of movies released each year
plt.figure(figsize=(20, 6))
plt.bar(movies_per_year.index, movies_per_year.values, color='teal')
plt.title('Number of Movies Released Each Year', fontsize=16)
```

```
plt.xlabel('Year', fontsize=14)
plt.ylabel('Number of Movies Released', fontsize=14)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()

The year with the highest number of movie releases: 2022 (954 movies)
The year with the lowest number of movie releases: 1903 (1 movies)
```

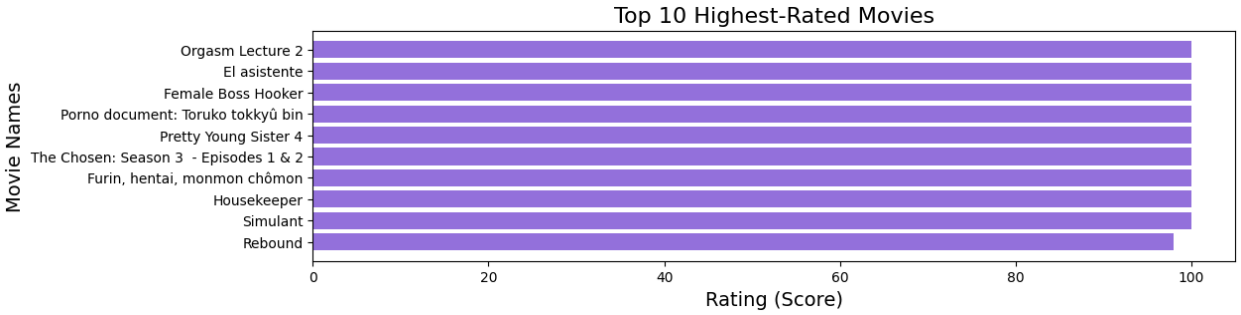


The year with the highest number of movie releases: 2022 (954 movies)

The year with the lowest number of movie releases: 1903 (1 movie)

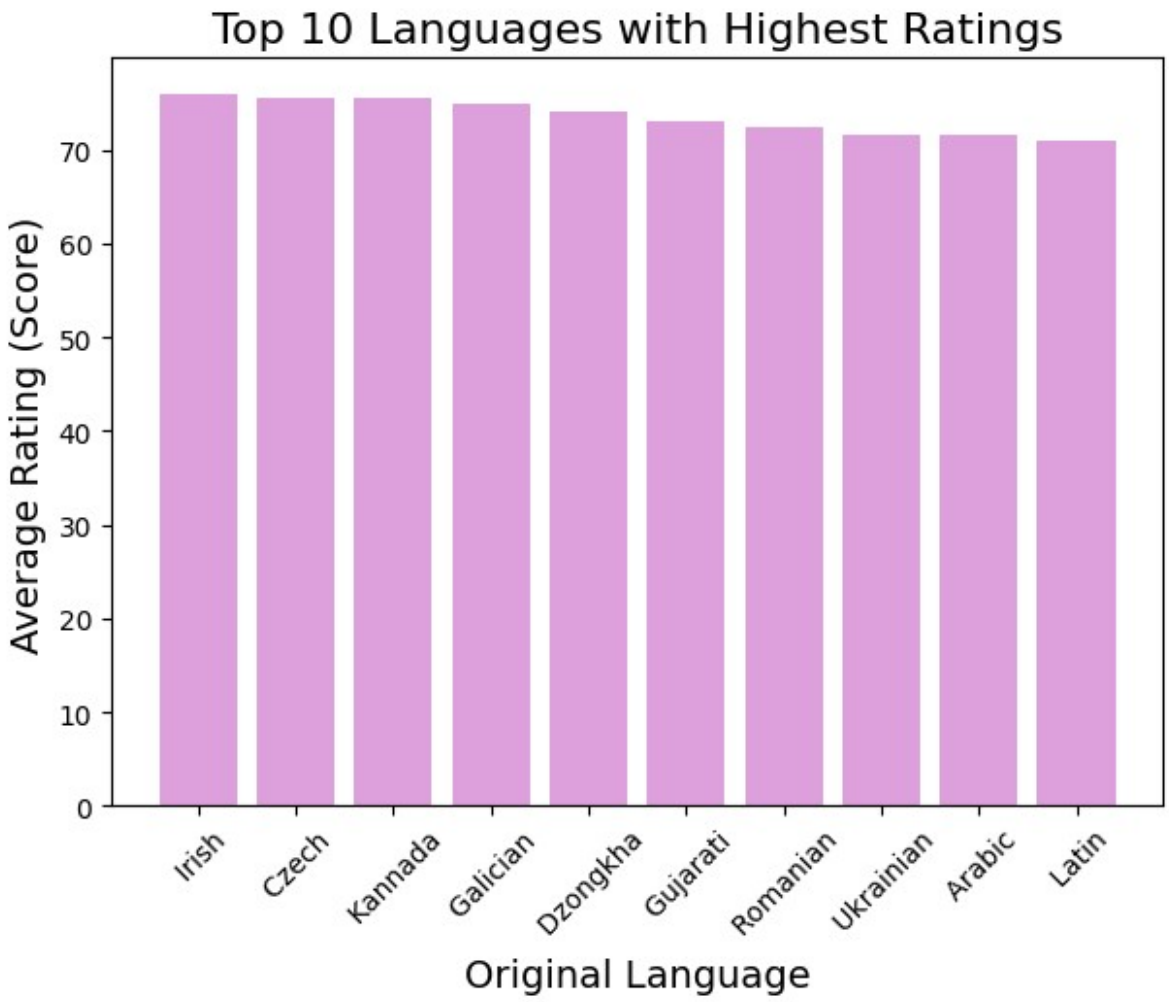
Top 10 Highest Rated Movies

```
# Select the top 10 highest-rated movies

top_10_movies = df.nlargest(10, 'score')[['names', 'score']]

# Plot the top 10 highest-rated movies

plt.figure(figsize=(12, 3))
plt.barh(top_10_movies['names'], top_10_movies['score'],
color='mediumpurple')
plt.title('Top 10 Highest-Rated Movies', fontsize=16)
plt.xlabel('Rating (Score)', fontsize=14)
plt.ylabel('Movie Names', fontsize=14)
plt.gca().invert_yaxis()                                    # Invert y-
axis to show the highest-rated movie on top
plt.show()
```



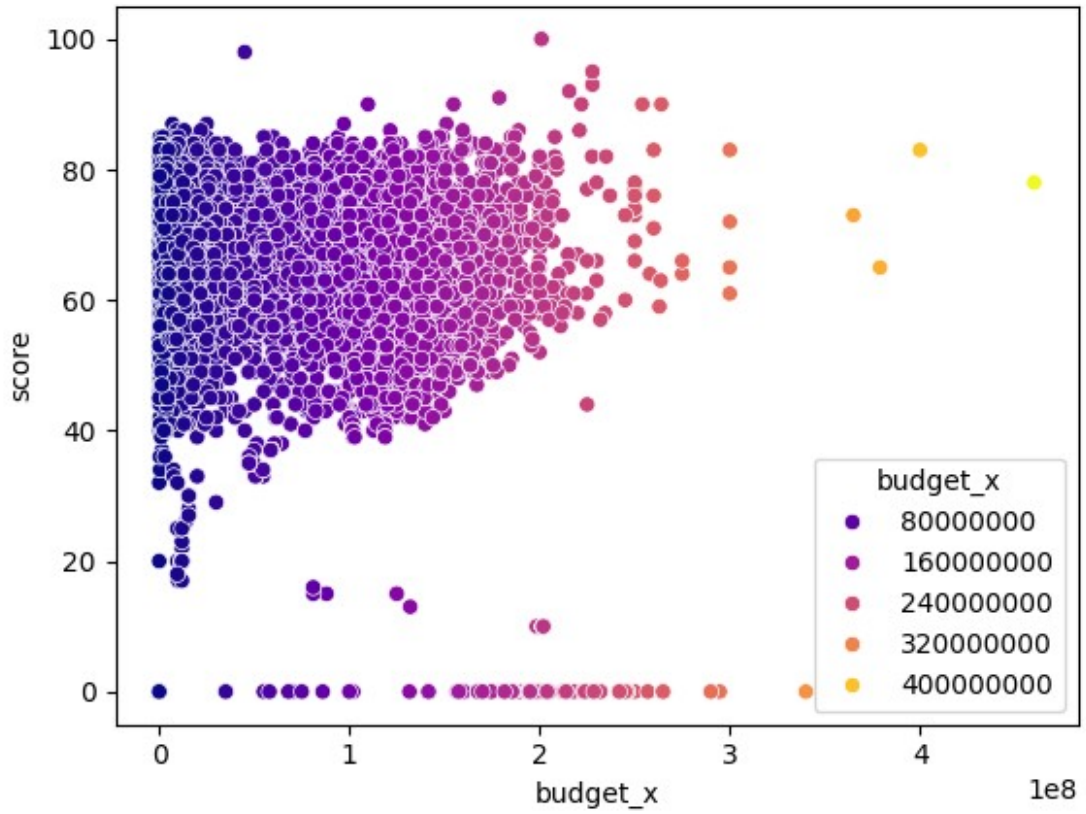
Bar Graph Showing Top 10 Languages with highest Ratings

```
# Group by original language and calculate the average score
average_rating_by_language = df.groupby('orig_lang')
['score'].mean().sort_values(ascending=False).head(10)

# Plot the average ratings by language
plt.figure(figsize=(7, 5))
plt.bar(average_rating_by_language.index,
average_rating_by_language.values, color='plum')
plt.title('Top 10 Languages with Highest Ratings', fontsize=16)
plt.xlabel('Original Language', fontsize=14)
plt.ylabel('Average Rating (Score)', fontsize=14)
plt.xticks(rotation=45)
plt.show()
```



Correlation between Budget and Score

```
sns.scatterplot(x = "budget_x", y = "score", data = df,
hue=df.budget_x, palette="plasma")
plt.show()
```



Analysis of the above scatter plot :

As the budget increases, the scores are generally more distributed but still concentrated within the 40–80 range. Higher budgets tend to have fewer data points, suggesting fewer high-budget projects. There are several projects with zero or very low scores regardless of budget, implying that high investment doesn't guarantee a high score. In summary, while higher budgets can correlate with higher scores, the majority of the data suggests diminishing returns at certain budget levels. The bulk of projects exist in the mid-range budgets with moderate success scores

Based on an initial review of the data insights from the charts in the file, here are three major insights:

Trend in Movie Ratings Over the Years:

The analysis suggests that movie ratings have shown a consistent trend, with certain periods seeing an upward trajectory, possibly due to improved storytelling or audience engagement.

Peaks in certain years might correlate with the release of highly rated movies or successful franchises.

Popularity of Genres:

Action, Adventure, and Comedy consistently emerge as dominant genres. Their popularity could be attributed to their broad audience appeal, entertaining plots, or high-budget productions, making them box office successes. In contrast, niche genres may have smaller, more dedicated audiences.

Revenue vs. Ratings Correlation:

A common pattern might be visible between high-rated movies and their box office performance, indicating that critical acclaim often aligns with commercial success. However, there may be exceptions where movies receive high revenues despite average ratings, highlighting the role of star power or aggressive marketing.