In [19]:
```
import numpy as np
import pandas as pd

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

Top 200 Movies based on IMDB and No. of Reviews

In [20]:
```
movie_df = pd.read_csv('movie_metadata.csv')
movie_df.head()
```

Out[20]:

	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name	a
0	Color	James Cameron	723.0	178.0	0.0	855.0	Joel David Moore	
1	Color	Gore Verbinski	302.0	169.0	563.0	1000.0	Orlando Bloom	
2	Color	Sam Mendes	602.0	148.0	0.0	161.0	Rory Kinnear	
3	Color	Christopher Nolan	813.0	164.0	22000.0	23000.0	Christian Bale	
4	NaN	Doug Walker	NaN	NaN	131.0	NaN	Rob Walker	

5 rows × 28 columns

In [21]:
```
movie_df.shape
```

Out[21]:
```
(5043, 28)
```

In [22]:
```
movie_df.columns
```

Out[22]:
```
Index(['color', 'director_name', 'num_critic_for_reviews', 'duration',
       'director_facebook_likes', 'actor_3_facebook_likes', 'actor_2_name',
       'actor_1_facebook_likes', 'gross', 'genres', 'actor_1_name',
       'movie_title', 'num_voted_users', 'cast_total_facebook_likes',
       'actor_3_name', 'facenumber_in_poster', 'plot_keywords',
       'movie_imdb_link', 'num_user_for_reviews', 'language', 'country',
       'content_rating', 'budget', 'title_year', 'actor_2_facebook_likes',
       'imdb_score', 'aspect_ratio', 'movie_facebook_likes'],
      dtype='object')
```

In [25]:
```
null_values = movie_df['num_user_for_reviews'].isnull().sum()
null_values
```

Out[25]:
```
21
```

In [28]:
```
movie_df_cleaned = movie_df.dropna(subset=['num_user_for_reviews'], how='all')
movie_df_cleaned.shape
```

Out[28]:
```
(5022, 28)
```

In [35]:
```
top200_movie_df = movie_df_cleaned.sort_values(by=['imdb_score', 'num_user_for_reviews'], ascending

relevant_columns = [
    'movie_title', 'imdb_score', 'num_user_for_reviews',
    'director_name', 'actor_1_name', 'actor_2_name', 'actor_3_name',
    'duration', 'genres', 'language', 'country', 'content_rating',
    'budget', 'title_year', 'gross', 'num_critic_for_reviews', 'plot_keywords'
]
result_df = top_200_movies[relevant_columns]
result_df
```

Out[35]:

	movie_title	imdb_score	num_user_for_reviews	director_name	actor_1_name	actor_2_name	actor_3_name	duration
270	The Lord of the Rings: The Fellowship of the R...	8.8	5060.0	Peter Jackson	Christopher Lee	Orlando Bloom	Billy Boyd	171.0
66	The Dark Knight	9.0	4667.0	Christopher Nolan	Christian Bale	Heath Ledger	Morgan Freeman	152.0
1937	The Shawshank Redemption	9.3	4144.0	Frank Darabont	Morgan Freeman	Jeffrey DeMunn	Bob Gunton	142.0
654	The Matrix	8.7	3646.0	Lana Wachowski	Keanu Reeves	Marcus Chong	Gloria Foster	136.0
339	The Lord of the Rings: The Return of the King	8.9	3189.0	Peter Jackson	Orlando Bloom	Billy Boyd	Bernard Hill	192.0
...	...	...	...	...	...	...	...	...
4444	Growing Up Smith	8.2	1.0	Frank Lotito	Brighton Sharbino	Jake Busey	Tim Guinee	102.0
2765	Towering Inferno	9.5	NaN	John Blanchard	Martin Short	Andrea Martin	Joe Flaherty	65.0
4372	A Beginner's Guide to Snuff	8.7	NaN	Mitchell Altieri	Kimberley Crossman	Luke Edwards	Bree Williamson	87.0
3816	Running Forever	8.6	NaN	Mike Mayhall	David Raizor	Cody Howard	Martin Kove	88.0
4945	The Brain That Sings	8.2	NaN	Amal Al-Agroobi	NaN	NaN	NaN	62.0

200 rows × 17 columns

Top Foreign Movies

In [36]:
```
foreign_movies_df = movie_df_cleaned[movie_df_cleaned['country'] != 'USA']

# Sort the foreign movies based on IMDb scores in descending order
top_foreign_movies = foreign_movies_df.sort_values(by='imdb_score', ascending=False)

# Display relevant columns for the top foreign movies
relevant_columns_foreign = [
    'movie_title', 'imdb_score', 'num_user_for_reviews',
    'director_name', 'actor_1_name', 'actor_2_name', 'actor_3_name',
    'duration', 'genres', 'language', 'country', 'content_rating',
    'budget', 'title_year', 'gross', 'num_critic_for_reviews', 'plot_keywords'
]

result_foreign_df = top_foreign_movies[relevant_columns_foreign]
result_foreign_df
```

Out[36]:

	movie_title	imdb_score	num_user_for_reviews	director_name	actor_1_name	actor_2_name	actor_3_name	duratio
2824	Dekalog	9.1	37.0	NaN	Krystyna Janda	Olaf Lubaszenko	Olgierd Lukaszewicz	55.
3207	Dekalog	9.1	37.0	NaN	Krystyna Janda	Olaf Lubaszenko	Olgierd Lukaszewicz	55.
4498	The Good, the Bad and the Ugly	8.9	780.0	Sergio Leone	Clint Eastwood	Luigi Pistilli	Enzo Petito	142.
270	The Lord of the Rings: The Fellowship of the R...	8.8	5060.0	Peter Jackson	Christopher Lee	Orlando Bloom	Billy Boyd	171.
4029	City of God	8.7	749.0	Fernando Meirelles	Alice Braga	Seu Jorge	Alexandre Rodrigues	135.
...	...	...	...	...	...	...	...	.
1072	Inchon	2.7	16.0	Terence Young	Laurence Olivier	Ben Gazzara	Jacqueline Bisset	140.
4525	The Vatican Exorcisms	2.6	10.0	Joe Marino	Piero Maggiò	Anella Vastola	Joe Marino	76.
2313	Alone in the Dark	2.3	521.0	Uwe Boll	Catherine Lough Haggquist	Darren Shahlavi	Karin Konoval	94.
1729	United Passions	2.0	22.0	Frédéric Auburtin	Fisher Stevens	Thomas Kretschmann	Jemima West	110.
2295	Superbabies: Baby Geniuses 2	1.9	129.0	Bob Clark	Scott Baio	Vanessa Angel	Peter Wingfield	88.

1228 rows × 17 columns

Top 10 Directors

In [37]:
```
# Group by director_name and calculate the mean IMDb score
directors_average_score = movie_df_cleaned.groupby('director_name')['imdb_score'].mean()

# Sort the directors based on average IMDb scores in descending order
top_10_directors = directors_average_score.sort_values(ascending=False).head(10)
top_10_directors
```

Out[37]:
```
director_name
Cary Bell            8.700
Sadyk Sher-Niyaz     8.700
Charles Chaplin      8.600
Ron Fricke           8.500
Damien Chazelle      8.500
Raja Menon           8.500
Majid Majidi         8.500
Sergio Leone         8.475
Tony Kaye            8.450
Christopher Nolan    8.425
Name: imdb_score, dtype: float64
```

Recommending Movies Based on Actors

In [38]:
```
# Create a list of unique actors
all_actors = set(movie_df_cleaned['actor_1_name'].unique())
all_actors.update(movie_df_cleaned['actor_2_name'].unique())
all_actors.update(movie_df_cleaned['actor_3_name'].unique())

all_actors = list(all_actors)
```

In [39]:
```
all_actors
```

```
 'Lindsay MacDonald',
 'Kane Hodder',
 'Tom Stedham',
 'Archie Panjabi',
 'Dylan Schombing',
 'Vondie Curtis-Hall',
 'Mario Cimarro',
 "Sofia Black-D'Elia",
 'Saul Stein',
 'Gabriel Reyes',
 'Alice Evans',
 'John Billingsley',
 'Jason Burkey',
 'Jeff Ament',
 'Scott Thompson',
 'Katharine McPhee',
 'Paul Michael Glaser',
 'Gary Daniels',
 'Michael Richards',
 'David Gallagher',
```

In [50]:
```
def recommend_movies_by_actor(actor_name, num_recommendations=5):
    # Filter movies that feature the specified actor in any of the three actor columns
    actor_movies = movie_df_cleaned[
        (movie_df_cleaned['actor_1_name'] == actor_name) |
        (movie_df_cleaned['actor_2_name'] == actor_name) |
        (movie_df_cleaned['actor_3_name'] == actor_name)
    ]

    # Sort the movies by IMDb score in descending order
    actor_movies_sorted = actor_movies.sort_values(by='imdb_score', ascending=False)

    # Exclude the movies that the specified actor starred in
    recommended_movies = actor_movies_sorted[~actor_movies_sorted['movie_title'].duplicated(keep='f

    return recommended_movies[['movie_title', 'imdb_score', 'genres', 'actor_1_name', 'actor_2_name
```

In [51]:
```
# Example: Get movie recommendations for a specific actor
actor_name = 'Tom Hanks'
recommendations = recommend_movies_by_actor(actor_name, num_recommendations=5)

# Display the recommendations
print(f"Top 5 Movie Recommendations for {actor_name}:")
recommendations
```

```
Top 5 Movie Recommendations for Tom Hanks:
```

Out[51]:

	movie_title	imdb_score	genres	actor_1_name	actor_2_name
836	Forrest Gump	8.8	Comedy|Drama	Tom Hanks	Siobhan Fallon Hogan
648	Saving Private Ryan	8.6	Action|Drama|War	Tom Hanks	Vin Diesel
712	The Green Mile	8.5	Crime|Drama|Fantasy|Mystery	Tom Hanks	Jeffrey DeMunn
1588	Toy Story	8.3	Adventure|Animation|Comedy|Family|Fantasy	Tom Hanks	John Ratzenberger
43	Toy Story 3	8.3	Adventure|Animation|Comedy|Family|Fantasy	Tom Hanks	John Ratzenberger

In [52]:
```
# Example: Get movie recommendations for a specific actor
actor_name = 'Christopher Lee'
recommendations = recommend_movies_by_actor(actor_name, num_recommendations=5)

# Display the recommendations
print(f"Top 5 Movie Recommendations for {actor_name}:")
recommendations
```

```
Top 5 Movie Recommendations for Christopher Lee:
```

Out[52]:

	movie_title	imdb_score	genres	actor_1_name	actor_2_name
270	The Lord of the Rings: The Fellowship of the R...	8.8	Action|Adventure|Drama|Fantasy	Christopher Lee	Orlando Bloom
340	The Lord of the Rings: The Two Towers	8.7	Action|Adventure|Drama|Fantasy	Christopher Lee	Orlando Bloom
236	Star Wars: Episode III - Revenge of the Sith	7.6	Action|Adventure|Fantasy|Sci-Fi	Natalie Portman	Christopher Lee
70	Hugo	7.5	Adventure|Drama|Family|Mystery	Chloë Grace Moretz	Christopher Lee
1642	Corpse Bride	7.4	Animation|Drama|Family|Fantasy|Musical|Romance	Johnny Depp	Christopher Lee

In [ ]:
```

```