A Data Analysis of IMDB and Rotten Tomatoes

Author: Brittney Nitta-Lee

Introduction

This project analyzes data from IMDB and Rotten Tomatoes to explore which genres and movie studios are the most successful in the movie industry. Microsoft is looking to create a new movie studio to produce original video content and wants to know what type of films are doing best at the box office. This analysis provides insights for the client to help decide what movies to create.

Questions

This project will address four questions about the movie industry: 1. What genre of movies are the most popular based on user ratings? 2. What are the top grossing movie genres? 3. What movie content rating based on genre is the most popular? 4. What are the top grossing movie studios?

Data

IMDB
The analysis examines a merged form of data from Box Office Mojo by IMDbPro and IMDB. BOM (Box Office Mojo) has data on domestic gross, and IMDB has data on movie genres.

Rotten Tomatoes
Rotten Tomatoes contains data on genres, user ratings and movie content rating that is used to address popular genres based on content rating and popular genres based on user ratings.

IMDB Data

The IMDB dataset has two tables that were used in this data cleaning. Movie Basics and Movie Ratings includes data files that provide movie genres, title, start year and average votes.

In [1]:
```
import sqlite3
import pandas as pd
```

In [2]:
```
conn = sqlite3.connect('im.db')
cur = conn.cursor()
```

Datacleaning

Pandas is used to format the IMDB data into the following dataframes.

In [14]:
```
movie_gross = pd.read_csv ('bom.movie_gross.csv')

movie_gross.head()
```

Out[14]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010

In [15]:
```
movie_basics = pd.read_sql("""
    SELECT *
    FROM movie_basics;
    """, conn)
movie_basics.head()
```

Out[15]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	do
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama	
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama	
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama	
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	

In [16]:
```
movie_ratings = pd.read_sql("""
    SELECT *
    FROM movie_ratings;
    """, conn)
movie_ratings.head()
```

Out[16]:

	movie_id	averagerating	numvotes
0	tt10356526	8.3	31
1	tt10384606	8.9	559
2	tt1042974	6.4	20
3	tt1043726	4.2	50352
4	tt1060240	6.5	21

Movie Basics and Movie Gross both have a column for individual movie titles. I want to see if the two datasets share the same datat under Title and Primary Title columns. I merged the datasets along the title and primary title columns will show movies that share the same title.

In [17]:
```
movie_basics = movie_gross.merge(movie_basics, how='inner', left_on='title', rig
movie_basics.head()
```

Out[17]:

	title_x	studio	domestic_gross_x	foreign_gross	year	movie_id	primary_title	original_titl
0	Toy Story 3	BV	415000000.0	652000000	2010	tt0435761	Toy Story 3	Toy Story
1	Inception	WB	292600000.0	535700000	2010	tt1375666	Inception	Inceptio
2	Shrek Forever After	P/DW	238700000.0	513900000	2010	tt0892791	Shrek Forever After	Shre Forever Afte
3	The Twilight Saga: Eclipse	Sum.	300500000.0	398000000	2010	tt1325004	The Twilight Saga: Eclipse	The Twiligh Saga: Eclips
4	Iron Man 2	Par.	312400000.0	311500000	2010	tt1228705	Iron Man 2	Iron Man

In [18]:
```
imbd_df= pd.read_sql("""

SELECT *
FROM movie_basics
JOIN movie_ratings
    USING(movie_id)
;
""", conn)
imbd_df.head()
```

Out[18]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	do
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama	
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama	
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama	
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	

In [19]:
```
Same_movie_titles = []
for title in imbd_df['primary_title'].unique():
    if title in movie_gross['title'].unique():
        Same_movie_titles.append(title)
```

In [20]:
```
print(Same_movie_titles[0:5])
len(Same_movie_titles)
```

```
['Wazir', 'On the Road', 'The Secret Life of Walter Mitty', 'A Walk Among the To
mbstones', 'Jurassic World']
```

Out[20]:
```
2598
```

I created a new dataframe to join Movie Basics and Movie ratings using Movie ID.

Since the genres are separated by commas, I wanted to split the genres up using the explode function.

In [11]:
```
imbd_df['genres'] = imbd_df['genres'].str.split(',')
imbd_genres_df = imbd_df.explode('genres')
```

Analysis

In [44]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

In [23]:
```
#explode function to separate genres
movie_basics['genres'] = movie_basics['genres'].str.split(',')
movie_basics_genres = movie_basics.explode('genres')
movie_basics_genres
```

Out[23]:

	title_x	studio	domestic_gross_x	foreign_gross	year	movie_id	primary_title	original
0	Toy Story 3	BV	415000000.0	652000000	2010	tt0435761	Toy Story 3	Toy S
0	Toy Story 3	BV	415000000.0	652000000	2010	tt0435761	Toy Story 3	Toy S
0	Toy Story 3	BV	415000000.0	652000000	2010	tt0435761	Toy Story 3	Toy S
1	Inception	WB	292600000.0	535700000	2010	tt1375666	Inception	Ince
1	Inception	WB	292600000.0	535700000	2010	tt1375666	Inception	Ince
...	...	...	...	...	...	...	...	
3363	Beauty and the Dogs	Osci.	8900.0	NaN	2018	tt6776572	Beauty and the Dogs	Aala Ka
3364	The Quake	Magn.	6200.0	NaN	2018	tt6523720	The Quake	Sk
3364	The Quake	Magn.	6200.0	NaN	2018	tt6523720	The Quake	Sk
3364	The Quake	Magn.	6200.0	NaN	2018	tt6523720	The Quake	Sk
3365	An Actor Prepares	Grav.	1700.0	NaN	2018	tt5718046	An Actor Prepares	An Pre

7471 rows × 13 columns

In [24]:
```
#group genres, domestic gross, studio
```

```
moviebasics_group_table = (
    movie_basics_genres
    .groupby('genres')
    .sum()
    .reset_index()
    .sort_values('domestic_gross_x', ascending = False)[['genres', 'domestic_gro
)
moviebasics_group_table
```

Out[24]:

	genres	domestic_gross_x
1	Adventure	4.191778e+10
0	Action	3.843915e+10
4	Comedy	3.249809e+10
7	Drama	3.105158e+10
17	Sci-Fi	1.495762e+10
19	Thriller	1.367092e+10
2	Animation	1.362289e+10
5	Crime	9.352542e+09
9	Fantasy	9.288773e+09
16	Romance	7.331809e+09
11	Horror	7.088680e+09
3	Biography	6.420383e+09
8	Family	5.597358e+09
6	Documentary	5.443313e+09
14	Mystery	4.974365e+09
10	History	2.943172e+09
18	Sport	2.122595e+09
12	Music	1.697182e+09
13	Musical	5.508563e+08
21	Western	5.294837e+08
20	War	2.814003e+08
15	News	2.184540e+07

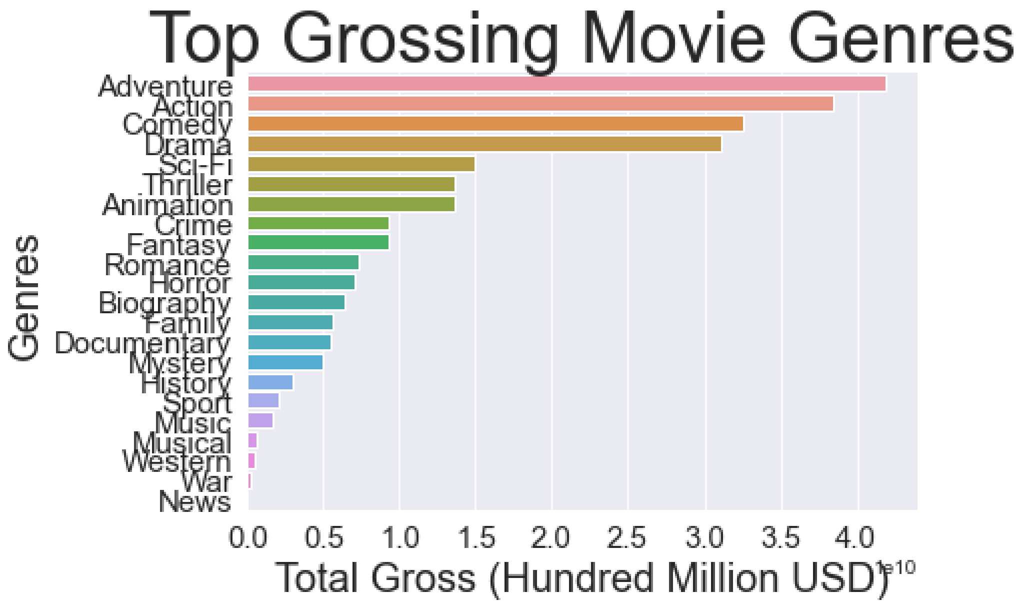
In [25]:

```
sns.set_style('darkgrid')
sns.set_palette('Set2')

sns.barplot(data=moviebasics_group_table, x="domestic_gross_x", y="genres", ci=N
sns.set(rc = {'figure.figsize':(15,15)})

plt.title('Top Grossing Movie Genres', fontsize=35, fontname='Arial')
plt.xlabel('Total Gross (Hundred Million USD)', fontsize=20, fontname='Arial')
plt.ylabel('Genres', fontsize=20, fontname='Arial')
plt.xticks(fontsize=15, fontname='Arial')
plt.yticks(fontsize=15, fontname='Arial')


sns.despine()
plt.show()
```



Rotten Tomatoes

I'll be using Pandas to open two Rotten Tomatoes datasets.

In [26]:
```
rt_reviews = pd.read_csv('rt.reviews.tsv', delimiter="\t", header=0, encoding="u
rt_reviews.head(6)
```

Out[26]:

	id	review	rating	fresh	critic	top_critic	publisher	date
0	3	A distinctly gallows take on contemporary fina...	3/5	fresh	PJ Nabarro	0	Patrick Nabarro	November 10, 2018
1	3	It's an allegory in search of a meaning that n...	NaN	rotten	Annalee Newitz	0	io9.com	May 23, 2018
2	3	... life lived in a bubble in financial dealin...	NaN	fresh	Sean Axmaker	0	Stream on Demand	January 4, 2018
3	3	Continuing along a line introduced in last yea...	NaN	fresh	Daniel Kasman	0	MUBI	November 16, 2017
4	3	... a perverse twist on neorealism...	NaN	fresh	NaN	0	Cinema Scope	October 12, 2017
5	3	... Cronenberg's Cosmopolis expresses somethin...	NaN	fresh	Michelle Orange	0	Capital New York	September 11, 2017

In [27]:
```
rt_info = pd.read_csv('rt.movie_info.tsv', delimiter = '\t', header=0, encoding=
rt_info.head()
```

Out[27]:

	id	synopsis	rating	genre	director	writer	theater_date	d

	id	synopsis	rating	genre	director	writer	theater_date	d
0	1	This gritty, fast-paced, and innovative police...	R	Action and Adventure|Classics|Drama	William Friedkin	Ernest Tidyman	Oct 9, 1971	
1	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	
2	5	Illeana Douglas delivers a superb performance ...	R	Drama|Musical and Performing Arts	Allison Anders	Allison Anders	Sep 13, 1996	
3	6	Michael Douglas runs afoul of a treacherous su...	R	Drama|Mystery and Suspense	Barry Levinson	Paul Attanasio|Michael Crichton	Dec 9, 1994	
4	7	NaN	NR	Drama|Romance	Rodney Bennett	Giles Cooper	NaN	

Both dataset contains different columns expect for one, which is id. I will inspect the id column to confirm that these id numbers correlate to the same movie.

In [28]:
```
rt_reviews['id'][990:999]
```

Out[28]:
```
990    25
991    25
992    25
993    25
994    25
995    25
996    25
997    25
998    25
Name: id, dtype: int64
```

In [29]:
```
rt_reviews['review'][995:1000]
```

Out[29]:
```
995    a respectful but inert advertisement for inter...
996    Ultimately, this vision of feudal Japan seems ...
997    Memo to Hollywood: Find another use for Keanu ...
998    The basics of the story remain unchanged, but ...
999    As impressive as these visual elements prove t...
Name: review, dtype: object
```

In [30]:
```
rt_info['synopsis'][18]
```

Out[30]:
```
"From ancient Japan's most enduring tale, the epic 3D fantasy-adventure 47 Ronin
is born. Keanu Reeves leads the cast as Kai, an outcast who joins Oishi (Hiroyuk
i Sanada), the leader of 47 outcast samurai. Together they seek vengeance upon t
```

```
he treacherous overlord who killed their master and banished their kind. To rest
ore honor to their homeland, the warriors embark upon a quest that challenges th
em with a series of trials that would destroy ordinary warriors. 47 Ronin is hel
med by visionary director Carl Erik Rinsch (The Gift). Inspired by styles as div
erse as Miyazaki and Hokusai, Rinsch will bring to life the stunning landscapes 
and enormous battles that will display the timeless Ronin story to global audien
ces in a way that's never been seen before. -- (C) Universal"
```

In [31]:
```
rt_info['id'][18]
```

Out[31]:
```
25
```

The movie ids are a match. Next I will merge the the movie id columns from both datasets.

In [32]:
```
#The movie IDs from RT_reviews and RT_info match so merge on ID
rotten_tomatoes_df = rt_info.merge(rt_reviews, how='inner', on='id')
rotten_toatoes_df.head(100)
```

Out[32]:

	id	synopsis	rating_x	genre	director	writer	theater_date	dvd_date	c
0	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	
1	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	
2	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	
3	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	
4	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	
...	...	...	...	...	...	...	...	...	
95	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	

	id	synopsis	rating_x	genre	director	writer	theater_date	dvd_date	c
96	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	
97	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	
98	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	
99	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	

100 rows × 19 columns

I created a new dataframe to reflect the columns that I need for my analysis.

In [33]:
```
rt_subset = rotten_tomatoes_df[['id', 'rating_x', 'genre', 'review', 'fresh']]
rt_subset.head()
```

Out[33]:

	id	rating_x	genre	review	fresh
0	3	R	Drama|Science Fiction and Fantasy	A distinctly gallows take on contemporary fina...	fresh
1	3	R	Drama|Science Fiction and Fantasy	It's an allegory in search of a meaning that n...	rotten
2	3	R	Drama|Science Fiction and Fantasy	... life lived in a bubble in financial dealin...	fresh
3	3	R	Drama|Science Fiction and Fantasy	Continuing along a line introduced in last yea...	fresh
4	3	R	Drama|Science Fiction and Fantasy	... a perverse twist on neorealism...	fresh

In [ ]:
```
#What genre has the most "fresh" review?
#need count of frequency for fresh values based on content rating and genre
#to count the number of fresh values I would need to create a for loop
```

Changed column names to format data.

In [34]:
```
rt_subset.rename(columns = {'fresh':'rating', 'rating_x':'contentrating'}, inpla
```

```
/Users/brittneynitta-lee/opt/anaconda3/envs/learn-env/lib/python3.8/site-package
s/pandas/core/frame.py:4296: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stab
le/user_guide/indexing.html#returning-a-view-versus-a-copy
  return super().rename(
```

I used the explode function to separate genres.

In [35]:
```
rt_subset['genre'] = rt_subset['genre'].str.split("|")
rt_subset_2 = rt_subset.explode('genre')
```

```
<ipython-input-35-0ca5509876af>:1: 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/stab
le/user_guide/indexing.html#returning-a-view-versus-a-copy
  rt_subset['genre'] = rt_subset['genre'].str.split("|")
```

I created a dictionary to assign numerical values to fresh and rotten ratings. Numerical values are now in a new column called numeric_rating.

In [36]:
```
rating_map = {'fresh': 1, 'rotten' : -1}
rt_subset_2['numeric_rating'] = rt_subset_2['rating'].map(rating_map)
rt_subset_2
```

Out[36]:

	id	contentrating	genre	review	rating	numeric_rating
0	3	R	Drama	A distinctly gallows take on contemporary fina...	fresh	1
0	3	R	Science Fiction and Fantasy	A distinctly gallows take on contemporary fina...	fresh	1
1	3	R	Drama	It's an allegory in search of a meaning that n...	rotten	-1
1	3	R	Science Fiction and Fantasy	It's an allegory in search of a meaning that n...	rotten	-1
2	3	R	Drama	... life lived in a bubble in financial dealin...	fresh	1
...	...	...	...	...	...	...
54431	2000	R	Action and Adventure	NaN	fresh	1
54431	2000	R	Art House and International	NaN	fresh	1
54431	2000	R	Comedy	NaN	fresh	1
54431	2000	R	Drama	NaN	fresh	1
54431	2000	R	Mystery and Suspense	NaN	fresh	1

120079 rows × 6 columns

In [37]:
```
genre_numeric_rating = (
    rt_subset_2
    .groupby('genre')
    .sum()
    .reset_index()
    .sort_values('numeric_rating', ascending = False)[['genre', 'numeric_rating'
)
genre_numeric_rating
```

Out[37]:

	genre	numeric_rating
8	Drama	10286
5	Comedy	3958
3	Art House and International	2334
15	Romance	2248
14	Mystery and Suspense	2002
0	Action and Adventure	1756
4	Classics	1153
12	Kids and Family	1119
1	Animation	929
16	Science Fiction and Fantasy	698
7	Documentary	612
13	Musical and Performing Arts	535
17	Special Interest	323
18	Sports and Fitness	183
20	Western	168
6	Cult Movies	38
10	Gay and Lesbian	30
9	Faith and Spirituality	19
2	Anime and Manga	7
19	Television	-65
11	Horror	-295

In [43]:
```
#save content rating as new df
content_numeric_rating = (
    rt_subset_2
    .groupby('numeric_rating')
    .sum()
    .reset_index()
    .sort_values('numeric_contentrating', ascending = False)[['numeric_rating',
)
content_numeric_rating
```

Out[43]:

	numeric_rating	numeric_contentrating
1	1	78911.0

	numeric_rating	numeric_contentrating
0	-1	51768.0

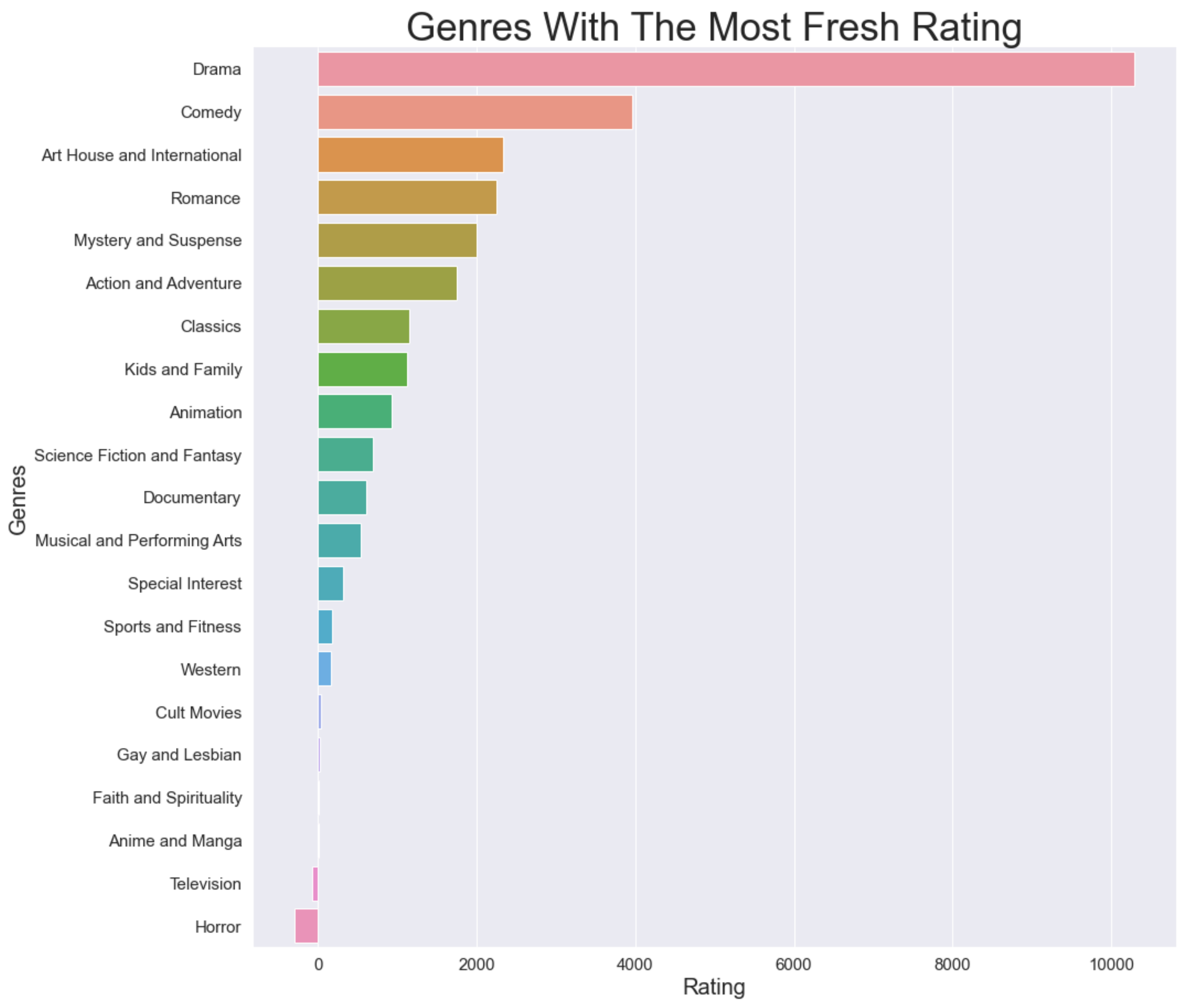
In [39]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [40]:
```
sns.set_style('darkgrid')
sns.set_palette('Set2')

sns.barplot(data=genre_numeric_rating, x="numeric_rating", y="genre", ci=None)
sns.set(rc = {'figure.figsize':(20,15)})

plt.title('Genres With The Most Fresh Rating', fontsize=35, fontname='Arial')
plt.xlabel('Rating', fontsize=20, fontname='Arial')
plt.ylabel('Genres', fontsize=20, fontname='Arial')
plt.xticks(fontsize=15, fontname='Arial')
plt.yticks(fontsize=15, fontname='Arial')


sns.despine()
plt.show()
```



To find the content rating with the most fresh reviews, I'll create a content rating dictionary and assign numeric values.

In [41]:
```
#What movie content rating has the most fresh reviews


contentrating_map = {'R': 0, 'PG' : 1, 'PG-13' : 2, 'NR' : 3, 'G' : 4}
rt_subset_2['numeric_contentrating'] = rt_subset_2['contentrating'].map(contentr
rt_subset_2
```

Out[41]:

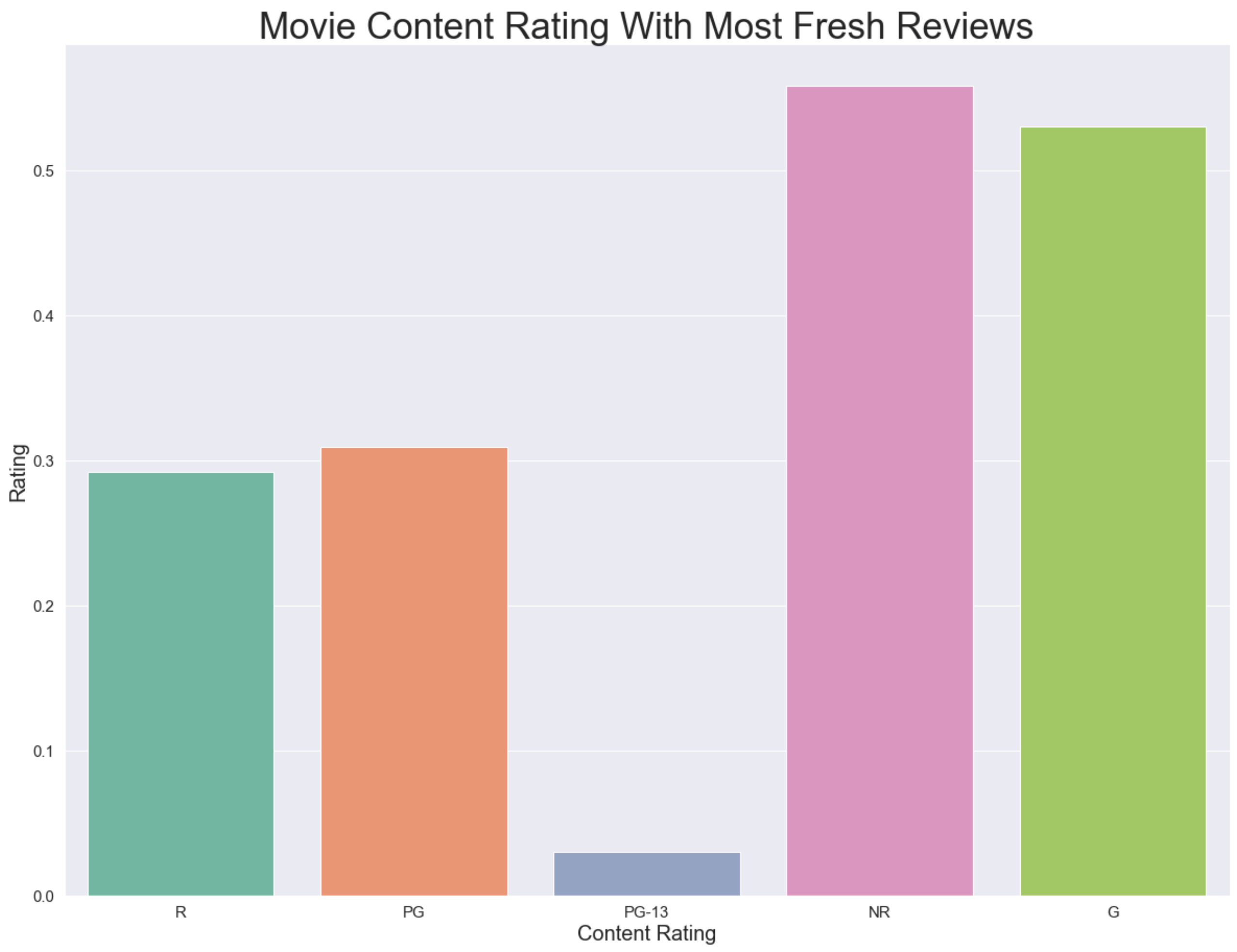
	id	contentrating	genre	review	rating	numeric_rating	numeric_contentra
0	3	R	Drama	A distinctly gallows take on contemporary fina...	fresh	1	
0	3	R	Science Fiction and Fantasy	A distinctly gallows take on contemporary fina...	fresh	1	
1	3	R	Drama	It's an allegory in search of a meaning that n...	rotten	-1	
1	3	R	Science Fiction and Fantasy	It's an allegory in search of a meaning that n...	rotten	-1	
2	3	R	Drama	... life lived in a bubble in financial dealin...	fresh	1	
...	...	...	...	...	...	...	
54431	2000	R	Action and Adventure	NaN	fresh	1	
54431	2000	R	Art House and International	NaN	fresh	1	
54431	2000	R	Comedy	NaN	fresh	1	
54431	2000	R	Drama	NaN	fresh	1	
54431	2000	R	Mystery and Suspense	NaN	fresh	1	

120079 rows × 7 columns

In [42]:
```
#What movie content rating has the most fresh reviews
sns.set_style('darkgrid')
sns.set_palette('Set2')
sns.barplot(data=rt_subset_2, x="contentrating", y="numeric_rating", ci=None)
sns.set(rc = {'figure.figsize':(15,15)})

plt.title('Movie Content Rating With Most Fresh Reviews', fontsize=35, fontname=
plt.xlabel('Content Rating', fontsize=20, fontname='Arial')
plt.ylabel('Rating', fontsize=20, fontname='Arial')
plt.xticks(fontsize=15, fontname='Arial')
plt.yticks(fontsize=15, fontname='Arial')


sns.despine()
plt.show()
```


Movie Content Rating With Most Fresh Reviews
Rating
R
PG
PG-13
NR
G
Content Rating