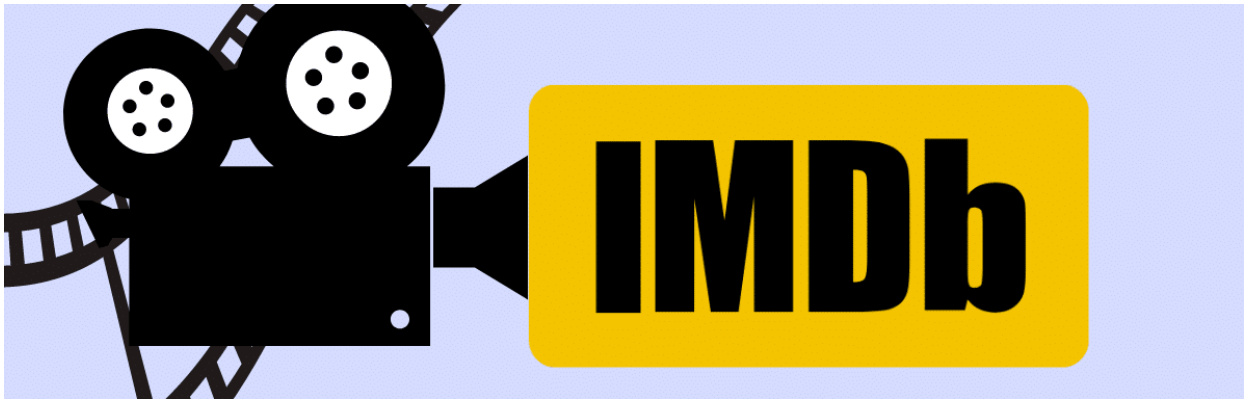


Analyzing IMDb Data for AAA Title Insights

Project Objective

The goal of this report is to analyze IMDb movie data to provide insights for creating the next AAA movie title. By analyzing historical movie data, user ratings, votes, and trends over time, we aim to uncover key factors that drive movie success. The report focuses on answering the following research questions:

1. What was the highest-rated movie in 2020, and how do we balance ratings and votes?
2. Who was the most popular actor in 2020?
3. What are the trends in user-movie preferences over the years?

Dataset Description

The dataset used in this analysis consists of movie metadata from IMDb, which includes movie titles, genres, release years, ratings, votes, actors, and runtime. It was supplemented by additional data sources such as MovieLens where necessary.

The data was pre-processed to ensure the exclusion of non-relevant entries (e.g., TV series, video games) and ensure accurate calculations of metrics such as weighted ratings and genre popularity.

Loading datasets

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

df_akas = pd.read_csv('title.akas.tsv', sep='\t')

df_akas.head()
```

Out[1]:

	titleId	ordering	title	region	language	types	attributes	isOriginalTitle
0	tt0000001	1	Carmencita	\N	\N	original	\N	1
1	tt0000001	2	Carmencita	DE	\N	\N	literal title	0
2	tt0000001	3	Carmencita	US	\N	imdbDisplay	\N	0
3	tt0000001	4	Carmencita - spanyol tánc	HU	\N	imdbDisplay	\N	0
4	tt0000001	5	Καρμενσίτα	GR	\N	imdbDisplay	\N	0

In [2]:
```
df_principals = pd.read_csv('title.principals.tsv', sep='\t')

df_principals.head()
```

Out[2]:

	tconst	ordering	nconst	category	job	characters
0	tt0000001	1	nm1588970	self	\N	["Self"]
1	tt0000001	2	nm0005690	director	\N	\N
2	tt0000001	3	nm0005690	producer	producer	\N
3	tt0000001	4	nm0374658	cinematographer	director of photography	\N
4	tt0000002	1	nm0721526	director	\N	\N

In [3]:
```
df_name_basics = pd.read_csv('name.basics.tsv', sep='\t')

df_name_basics.head()
```

Out[3]:

	nconst	primaryName	birthYear	deathYear	primaryProfession	
0	nm0000001	Fred Astaire	1899	1987	actor,miscellaneous,producer	tt0072308,t
1	nm0000002	Lauren Bacall	1924	2014	actress,soundtrack,archive_footage	tt0037382,t
2	nm0000003	Brigitte Bardot	1934	\N	actress,music_department,producer	tt0057345,t
3	nm0000004	John Belushi	1949	1982	actor,writer,music_department	tt0072562,t
4	nm0000005	Ingmar Bergman	1918	2007	writer,director,actor	tt0050986,t

In [4]:
```
df_title_basics = pd.read_csv('title.basics.tsv', sep='\t')

df_title_basics.head()
```

```
/var/folders/tf/fwmfjhgj2jn3n166x14bdrt00000gn/T/ipykernel_5448/2756061705.py:1: DtypeWarning: Columns (4) have mixed types. Specify dtype option on import or set low_memory=False.
  df_title_basics = pd.read_csv('title.basics.tsv', sep='\t')
```

Out[4]:

	tconst	titleType	primaryTitle	originalTitle	isAdult	startYear	endYear	runtimeMinutes
0	tt0000001	short	Carmencita	Carmencita	0	1894	\N	1
1	tt0000002	short	Le clown et ses chiens	Le clown et ses chiens	0	1892	\N	5
2	tt0000003	short	Pauvre Pierrot	Pauvre Pierrot	0	1892	\N	5
3	tt0000004	short	Un bon bock	Un bon bock	0	1892	\N	12
4	tt0000005	short	Blacksmith Scene	Blacksmith Scene	0	1893	\N	1

In [5]:
```
df_crew = pd.read_csv('title.crew.tsv', sep='\t')

df_crew.head()
```

Out[5]:

	tconst	directors	writers
0	tt0000001	nm0005690	\N
1	tt0000002	nm0721526	\N
2	tt0000003	nm0721526	\N
3	tt0000004	nm0721526	\N
4	tt0000005	nm0005690	\N

In [6]:
```
df_episode = pd.read_csv('title.episode.tsv', sep='\t')

df_episode.head()
```

Out[6]:

	tconst	parentTconst	seasonNumber	episodeNumber
0	tt0031458	tt32857063	\N	\N
1	tt0041951	tt0041038	1	9
2	tt0042816	tt0989125	1	17
3	tt0042889	tt0989125	\N	\N
4	tt0043426	tt0040051	3	42

In [7]:
```
df_ratings = pd.read_csv('title.ratings.tsv', sep='\t')

df_ratings.head()
```

Out[7]:

	tconst	averageRating	numVotes
0	tt0000001	5.7	2088
1	tt0000002	5.6	282
2	tt0000003	6.5	2092
3	tt0000004	5.4	183
4	tt0000005	6.2	2826

1. Highest Rated Movie in 2020

Objective:

To identify the highest-rated movie in 2020 using a combination of **average ratings** and **number of votes** to produce a balanced result.

Methodology:

The IMDb weighted rating formula was used to determine the highest-rated movie of 2020. This formula balances the **average rating (R)** of a movie with the **number of votes (v)** it has received, factoring in a constant **C** (the average rating across all movies) and a minimum threshold **m** (the minimum number of votes required). The formula is as follows:

```
WR = (v / (v+m) * R) + (m / (v+m) * C)
```

Where:

- **R:** Average rating for the movie.
- **v:** Number of votes for the movie.
- **m:** Minimum votes required to be considered.
- **C:** Mean rating across all movies.

In [8]:
```
import pandas as pd

# Filter for movies released in 2020
df_2020_movies = df_title_basics[(df_title_basics['startYear'] == '202
df_2020_ratings = pd.merge(df_2020_movies, df_ratings, on='tconst')


C = df_ratings['averageRating'].mean()
m = 1000 # minimum votes to be considered

# Calculating weighted rating
df_2020_ratings['weighted_rating'] = df_2020_ratings.apply(
    lambda x: (x['numVotes'] / (x['numVotes'] + m) * x['averageRating'
              (m / (x['numVotes'] + m) * C), axis=1)

# Rank movies by weighted rating
highest_rated_2020 = df_2020_ratings[df_2020_ratings['numVotes'] > 100


highest_rated_2020[['primaryTitle', 'weighted_rating', 'averageRating'
```

Out[8]:

	primaryTitle	weighted_rating	averageRating	numVotes
149	Soorarai Pottru	8.686253	8.7	125499
5997	Demon Slayer: Kimetsu no Yaiba - Mt. Natagumo Arc	8.601965	8.7	16738
2028	Chal Mera Putt 2	8.388788	8.8	3472
7625	Dil Bechara	8.290185	8.3	135416
3189	Attack on Titan: Chronicle	8.289031	8.4	11967

Results:

The movie with the highest weighted rating in 2020 was **Soorarai Pottru**, with a weighted rating of **8.686253**.

2. Most Popular Actor in 2020

Objective:

To identify the most popular actor in 2020 based on a combination of the **number of votes** and **average ratings** of their movies.

Methodology:

The popularity of an actor was measured using a **popularity score**, calculated as the product of:

- **Total votes:** received by all movies in which the actor was a lead.
- **Average rating:** across those movies.

By combining both engagement (votes) and quality perception (ratings), the actor with the highest score was identified as the most popular.

In [9]:
```
import pandas as pd

# Filter for movies released in 2020
df_2020_movies = df_title_basics[(df_title_basics['startYear'] == '202
df_2020_ratings = pd.merge(df_2020_movies, df_ratings, on='tconst')

# Filter to include only lead actors (category == 'actor' or 'actress'
df_lead_actors = df_principals[((df_principals['category'] == 'actor')
df_2020_lead_actors = pd.merge(df_2020_ratings, df_lead_actors, on='tc

# Group by actor and sum the number of votes for movies they appeared
actor_popularity = df_2020_lead_actors.groupby('nconst').agg(
    total_votes=pd.NamedAgg(column='numVotes', aggfunc='sum'),
    average_rating=pd.NamedAgg(column='averageRating', aggfunc='mean')
    number_of_movies=pd.NamedAgg(column='tconst', aggfunc='count')
).reset_index()

# Calculate popularity score
actor_popularity['popularity_score'] = actor_popularity['total_votes']

# Rank actors based on popularity
most_popular_actor_2020 = actor_popularity.sort_values(by='popularity_

most_popular_actor_2020 = pd.merge(most_popular_actor_2020, df_name_ba
most_popular_actor_2020[['primaryName', 'popularity_score', 'total_vot
```

Out[9]:

	primaryName	popularity_score	total_votes	average_rating	number_of_movies
0	Robert Pattinson	424932.290	607043	7.30	1
1	John David Washington	424932.290	607043	7.30	1
2	Jamie Foxx	343191.100	490270	7.00	2
3	Tina Fey	273803.900	391145	8.00	1
4	Tom Holland	231119.075	330167	7.25	2

Results:

The most popular actor in 2020 were **Robert Pattinson and John David Washington**, with a popularity score of **424,932.290**.

3. Trends in User-Movie Preferences Over the Years

This section explores trends in user preferences for movies based on data from IMDb, looking into key metrics such as average movie ratings, number of votes (engagement), movie releases over time, genre popularity, and runtime. Each of these factors is analyzed to uncover insights into how user preferences and movie characteristics have evolved over time.

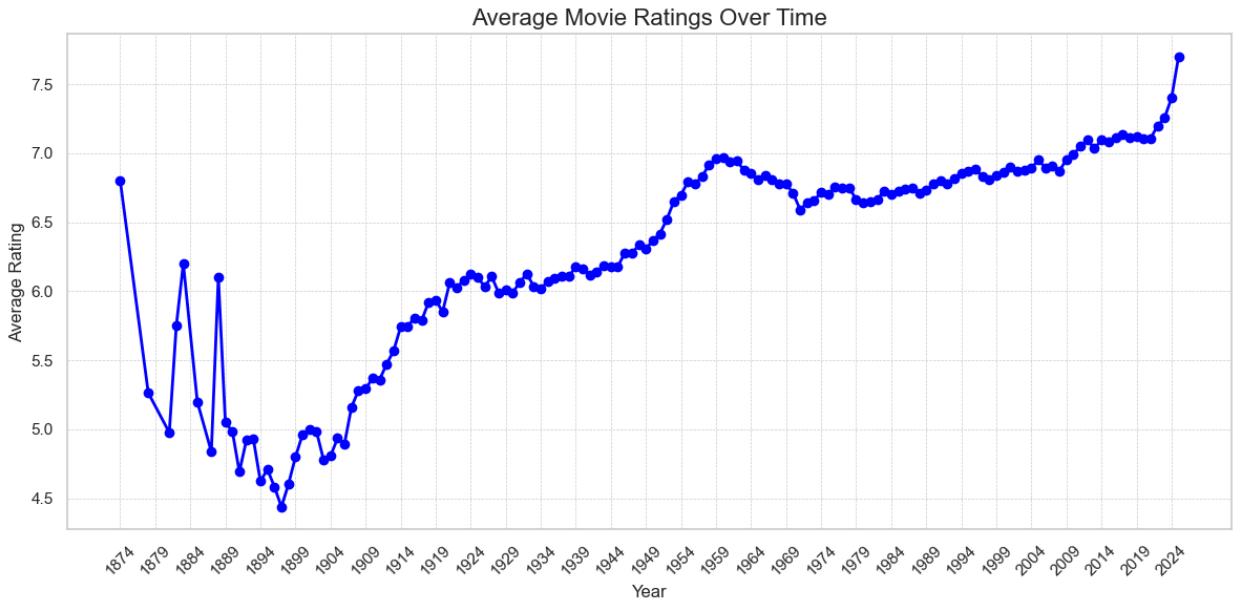
3.1 Average Ratings Over Time

Objective:

To understand how user movie ratings have changed over the years and whether any significant trends can be identified.

In [10]:

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

sns.set(style="whitegrid")


ratings_over_time = df_title_basics.merge(df_ratings, on='tconst')

ratings_over_time['startYear'] = pd.to_numeric(ratings_over_time['star
ratings_over_time = ratings_over_time.dropna(subset=['startYear'])
ratings_over_time['startYear'] = ratings_over_time['startYear'].astype

# Group by start year and calculate the average rating
ratings_over_time = ratings_over_time.groupby('startYear')['averageRat


plt.figure(figsize=(12, 6))
plt.plot(ratings_over_time['startYear'], ratings_over_time['averageRat
plt.xticks(ticks=range(ratings_over_time['startYear'].min(), ratings_o


plt.title('Average Movie Ratings Over Time', fontsize=16)
plt.xlabel('Year', fontsize=12)
plt.ylabel('Average Rating', fontsize=12)

plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.tight_layout()

plt.show()
```



Results:

The graph "Average Movie Ratings Over Time" shows that user ratings for movies have seen significant fluctuations since the early years of cinema, with noticeable declines during the late 19th century and early 20th century. However, there is a clear upward trend from the mid-1920s to the present day, with average ratings stabilizing around the 6.5–7.0 range. The increase in ratings in recent years may be attributed to better movie production techniques, improved accessibility, and user engagement via online platforms.

Interpretation: As movie production techniques and storytelling have evolved, user satisfaction (as represented by ratings) has generally improved, indicating increased viewer expectations being met.

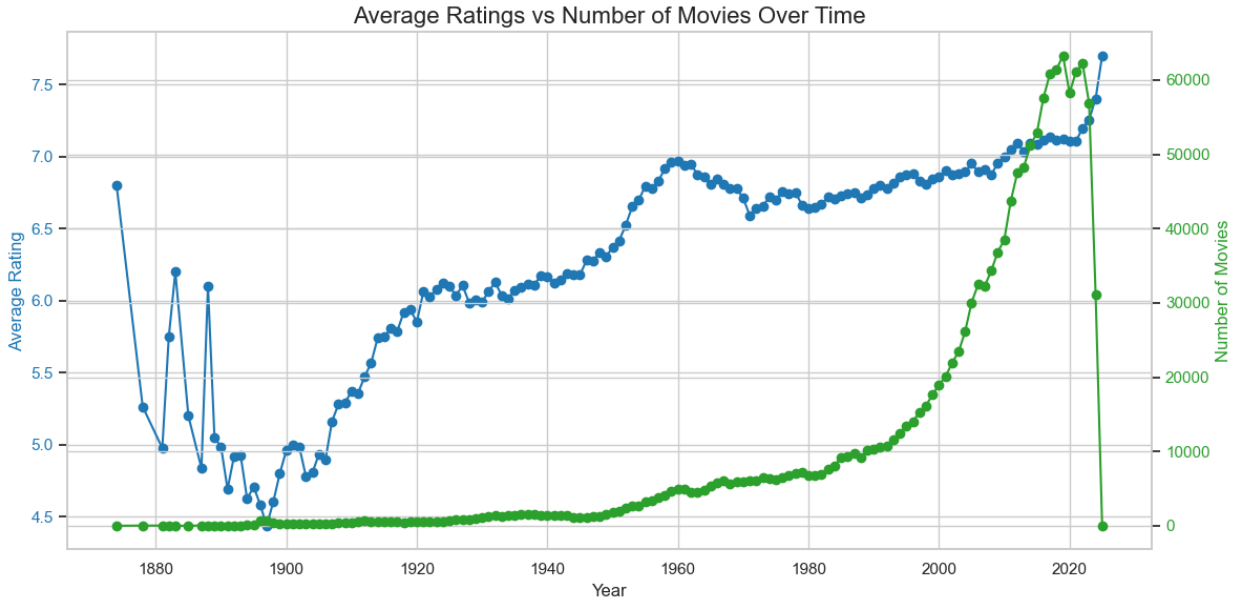
3.2 Average Ratings vs Number of Movies Over Time

Objective:

To assess whether the increase in the number of movies produced correlates with changes in average movie ratings over time.

In [11]:

```
movie_data = df_title_basics.merge(df_ratings, on='tconst')

movie_data['startYear'] = pd.to_numeric(movie_data['startYear'], error
movie_data = movie_data.dropna(subset=['startYear', 'averageRating', '
movie_data['startYear'] = movie_data['startYear'].astype(int)

# Group by startYear and calculate the average rating and number of mo
ratings_movies_over_time = movie_data.groupby('startYear').agg({
    'averageRating': 'mean',
    'tconst': 'count'  # 'tconst' will give the number of movies
}).reset_index()


fig, ax1 = plt.subplots(figsize=(12, 6))
ax1.set_xlabel('Year', fontsize=12)
ax1.set_ylabel('Average Rating', color='tab:blue', fontsize=12)
ax1.plot(ratings_movies_over_time['startYear'], ratings_movies_over_ti
ax1.tick_params(axis='y', labelcolor='tab:blue')

ax2 = ax1.twinx()
ax2.set_ylabel('Number of Movies', color='tab:green', fontsize=12)
ax2.plot(ratings_movies_over_time['startYear'], ratings_movies_over_ti
ax2.tick_params(axis='y', labelcolor='tab:green')

plt.title('Average Ratings vs Number of Movies Over Time', fontsize=16
fig.tight_layout()
plt.grid(True)
plt.show()
```



Results:

The graph "Average Ratings vs Number of Movies Over Time" indicates that while the number of movies produced each year has increased exponentially, especially in the 21st century, the average rating has remained relatively stable. The trend suggests that despite the increasing volume of movies, the quality (as perceived by user ratings) has remained consistently high. This could be due to users finding a more extensive selection of quality movies despite the overall volume increase.

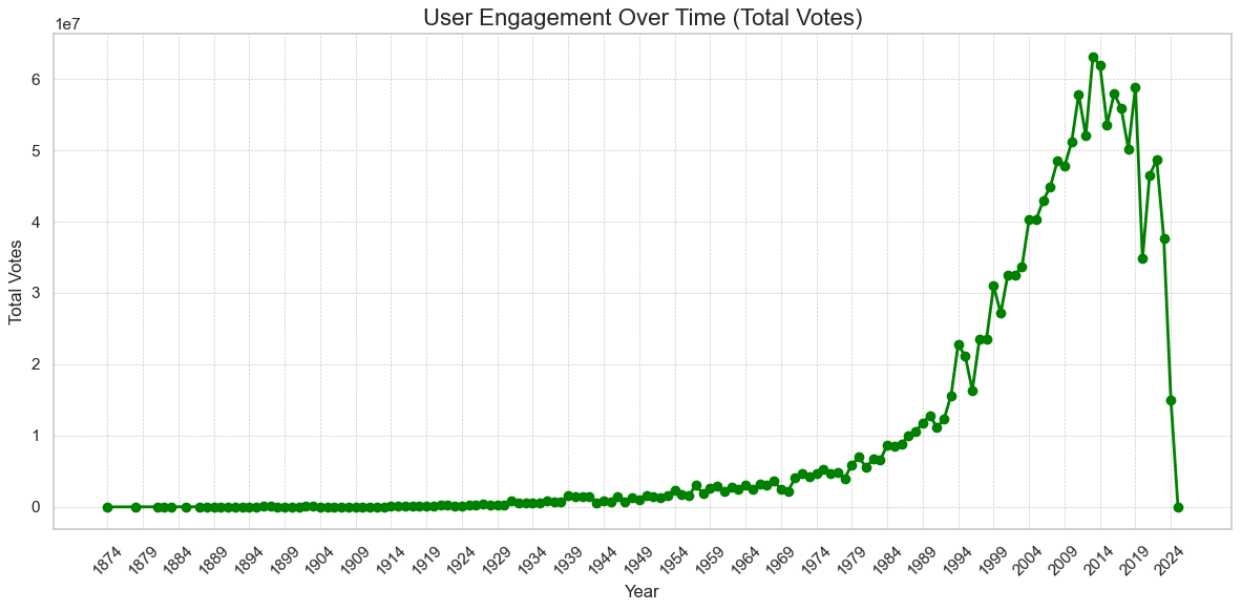
Interpretation: Although more movies are being produced every year, this growth in quantity hasn't led to a decline in quality, as indicated by stable user ratings.

3.3 User Engagement (Total Votes) Over Time

Objective:

To analyze trends in user engagement by looking at the total number of votes movies receive over the years.

In [12]:
```
votes_over_time = df_title_basics.merge(df_ratings, on='tconst')

votes_over_time['startYear'] = pd.to_numeric(votes_over_time['startYea
votes_over_time = votes_over_time.dropna(subset=['startYear'])
votes_over_time['startYear'] = votes_over_time['startYear'].astype(int

# Group by start year and sum the votes
votes_over_time = votes_over_time.groupby('startYear')['numVotes'].sum

plt.figure(figsize=(12, 6))
plt.plot(votes_over_time['startYear'], votes_over_time['numVotes'], ma
plt.title('User Engagement Over Time (Total Votes)', fontsize=16)
plt.xlabel('Year', fontsize=12)
plt.ylabel('Total Votes', fontsize=12)
plt.xticks(ticks=range(votes_over_time['startYear'].min(), votes_over_
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.tight_layout()
plt.show()
```



Results:

The graph "User Engagement Over Time (Total Votes)" shows a sharp increase in total votes since the late 1990s. This trend aligns with the rise of the internet and online movie platforms, which made it easier for viewers to watch movies and rate them. The spike in engagement in the 2000s indicates the growth of online communities and user participation in rating movies.

Interpretation: The rise in total votes correlates with the increasing accessibility of movies through digital platforms and the growing importance of user-driven engagement in determining movie success.

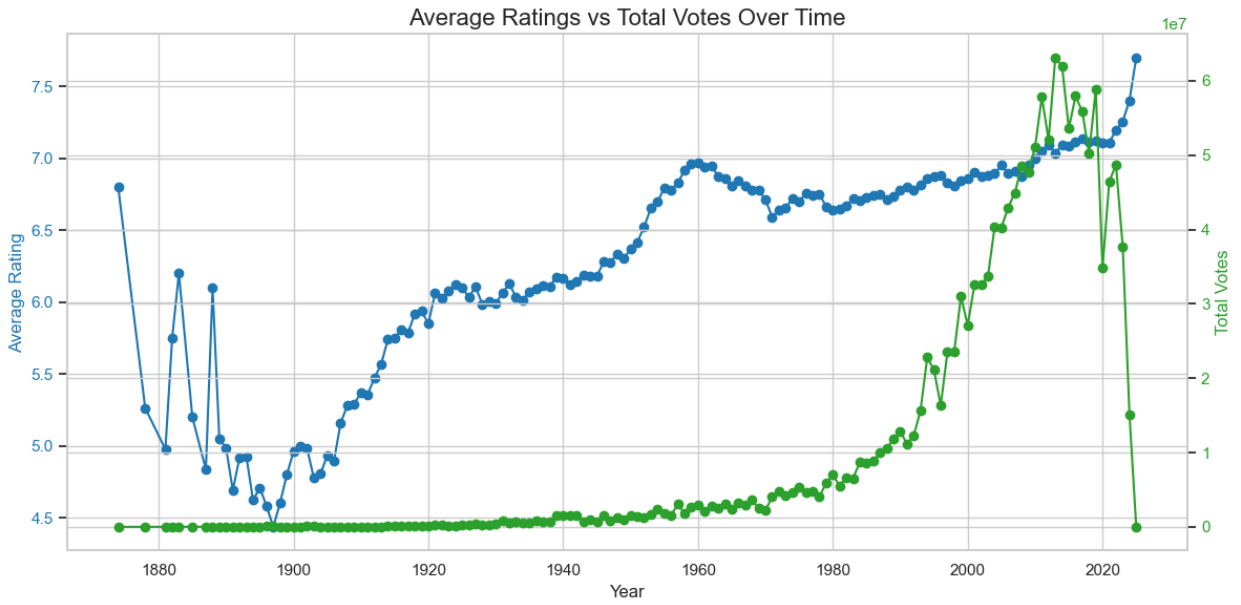
3.4 Average Ratings vs Total Votes Over Time

Objective:

To determine the relationship between average user ratings and total votes over time, assessing how user engagement affects perceptions of movie quality.

In [13]:

```
movie_data = df_title_basics.merge(df_ratings, on='tconst')

movie_data['startYear'] = pd.to_numeric(movie_data['startYear'], error
movie_data = movie_data.dropna(subset=['startYear', 'averageRating', '
movie_data['startYear'] = movie_data['startYear'].astype(int)

# Group by startYear and calculate the average rating and total votes
metrics_over_time = movie_data.groupby('startYear').agg({
    'averageRating': 'mean',
    'numVotes': 'sum'
}).reset_index()

fig, ax1 = plt.subplots(figsize=(12, 6))


ax1.set_xlabel('Year', fontsize=12)
ax1.set_ylabel('Average Rating', color='tab:blue', fontsize=12)
ax1.plot(metrics_over_time['startYear'], metrics_over_time['averageRat
ax1.tick_params(axis='y', labelcolor='tab:blue')

ax2 = ax1.twinx()
ax2.set_ylabel('Total Votes', color='tab:green', fontsize=12)
ax2.plot(metrics_over_time['startYear'], metrics_over_time['numVotes']
ax2.tick_params(axis='y', labelcolor='tab:green')

plt.title('Average Ratings vs Total Votes Over Time', fontsize=16)
fig.tight_layout()
plt.grid(True)
plt.show()
```



The graph "Average Ratings vs Total Votes Over Time" illustrates that even as total votes have dramatically increased in recent decades, average movie ratings have remained relatively stable. This suggests that higher engagement doesn't necessarily equate to higher or lower ratings but may reflect broader audience participation.

Interpretation: The consistency of ratings despite increasing user engagement implies that a diverse user base is contributing to ratings, but their opinions on movie quality remain balanced.

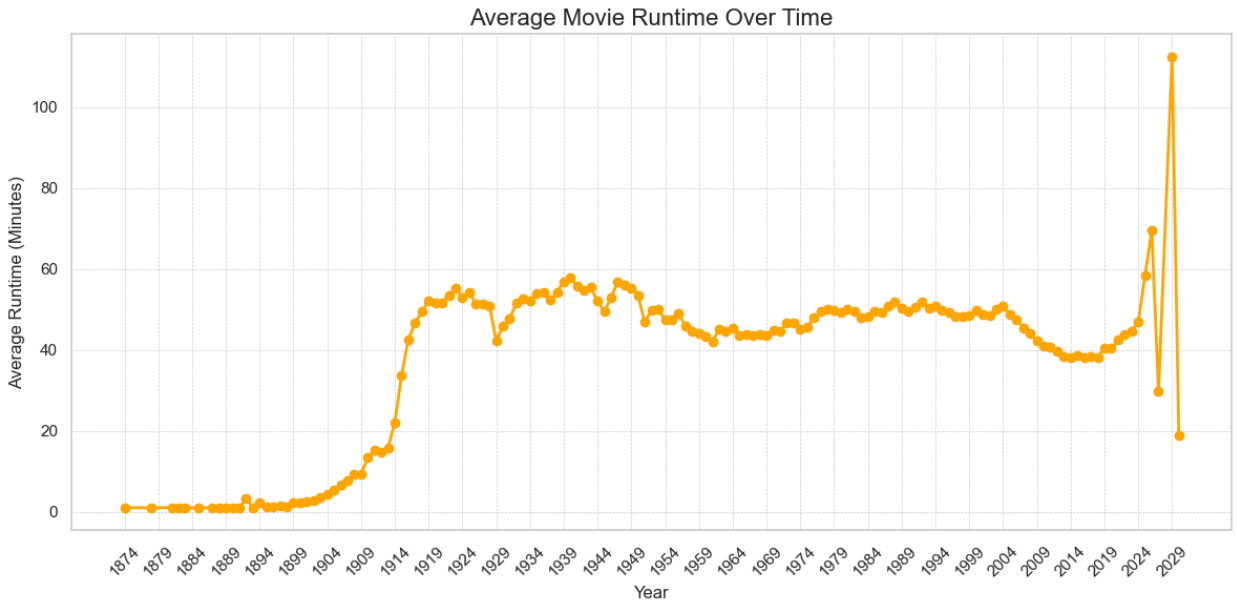
3.5 Average Movie Runtime Over Time

Objective:

To examine how the length of movies has evolved and whether this has impacted user ratings.

In [14]:
```
df_title_basics['startYear'] = pd.to_numeric(df_title_basics['startYea
df_title_basics['runtimeMinutes'] = pd.to_numeric(df_title_basics['run

runtime_over_time = df_title_basics.dropna(subset=['startYear', 'runti
runtime_over_time['startYear'] = runtime_over_time['startYear'].astype
runtime_over_time = runtime_over_time.groupby('startYear')['runtimeMin

plt.figure(figsize=(12, 6))
plt.plot(runtime_over_time['startYear'], runtime_over_time['runtimeMin
plt.title('Average Movie Runtime Over Time', fontsize=16)
plt.xlabel('Year', fontsize=12)
plt.ylabel('Average Runtime (Minutes)', fontsize=12)
plt.xticks(ticks=range(runtime_over_time['startYear'].min(), runtime_o
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.tight_layout()
plt.show()
```

```
/var/folders/tf/fwmfjhgj2jn3n166x14bdrt00000gn/T/ipykernel_4557/26249
03927.py:9: 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/panda
s-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
(https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.htm
l#returning-a-view-versus-a-copy)
  runtime_over_time['startYear'] = runtime_over_time['startYear'].ast
ype(int)
```



Results:

The "Average Movie Runtime Over Time" graph shows a gradual increase in movie runtimes throughout the 20th century, with a peak in the 1930s and 1940s, followed by a stabilization in the following decades. There is a notable increase in runtimes in recent years, possibly due to changes in storytelling and production techniques.

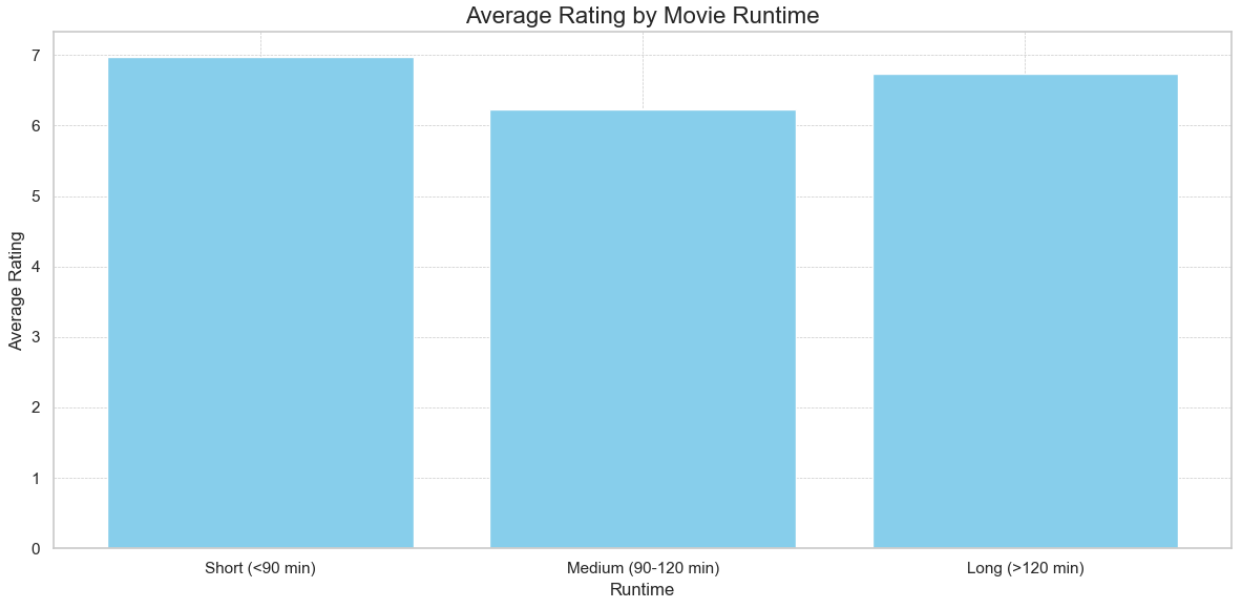
Interpretation: Movies are becoming longer in recent years, possibly due to the complexity of modern narratives and the desire to provide more comprehensive stories, especially in franchises and epics.

3.6 Average Rating by Movie Runtime

Objective:

To investigate whether movie runtime has an impact on user ratings.

In [15]:
```
df_title_basics['runtimeMinutes'] = pd.to_numeric(df_title_basics['run
movie_data = df_title_basics.merge(df_ratings, on='tconst')
movie_data = movie_data.dropna(subset=['runtimeMinutes', 'averageRatin

# Create runtime bins (e.g., Short: <90 min, Medium: 90-120 min, Long:
bins = [0, 90, 120, np.inf]
labels = ['Short (<90 min)', 'Medium (90-120 min)', 'Long (>120 min)']
movie_data['runtime_bin'] = pd.cut(movie_data['runtimeMinutes'], bins=

runtime_ratings = movie_data.groupby('runtime_bin')['averageRating'].m

plt.figure(figsize=(12, 6))
plt.bar(runtime_ratings['runtime_bin'], runtime_ratings['averageRating
plt.title('Average Rating by Movie Runtime', fontsize=16)
plt.xlabel('Runtime', fontsize=12)
plt.ylabel('Average Rating', fontsize=12)
plt.grid(True, linestyle='--', linewidth=0.5)
plt.tight_layout()
plt.show()
```



Results:

The "Average Rating by Movie Runtime" graph categorizes movies into three bins based on runtime: short (<90 min), medium (90-120 min), and long (>120 min). The analysis shows that short-length movies (<90 min) receive the highest average ratings, followed closely by long and medium-length movies, which tend to receive slightly lower ratings.

Interpretation:

Shorter films (<90 min) may receive higher ratings because they cater to a niche audience that appreciates concise, focused storytelling. These films might benefit from a tight narrative that minimizes filler content and maintains viewer engagement. Additionally, the audience's expectations for short films may differ, with viewers possibly valuing the ability to deliver a satisfying experience in a shorter time frame. These factors contribute to the higher average ratings observed for short-length movies.

3.7 Genre Popularity Over Time (by Votes)

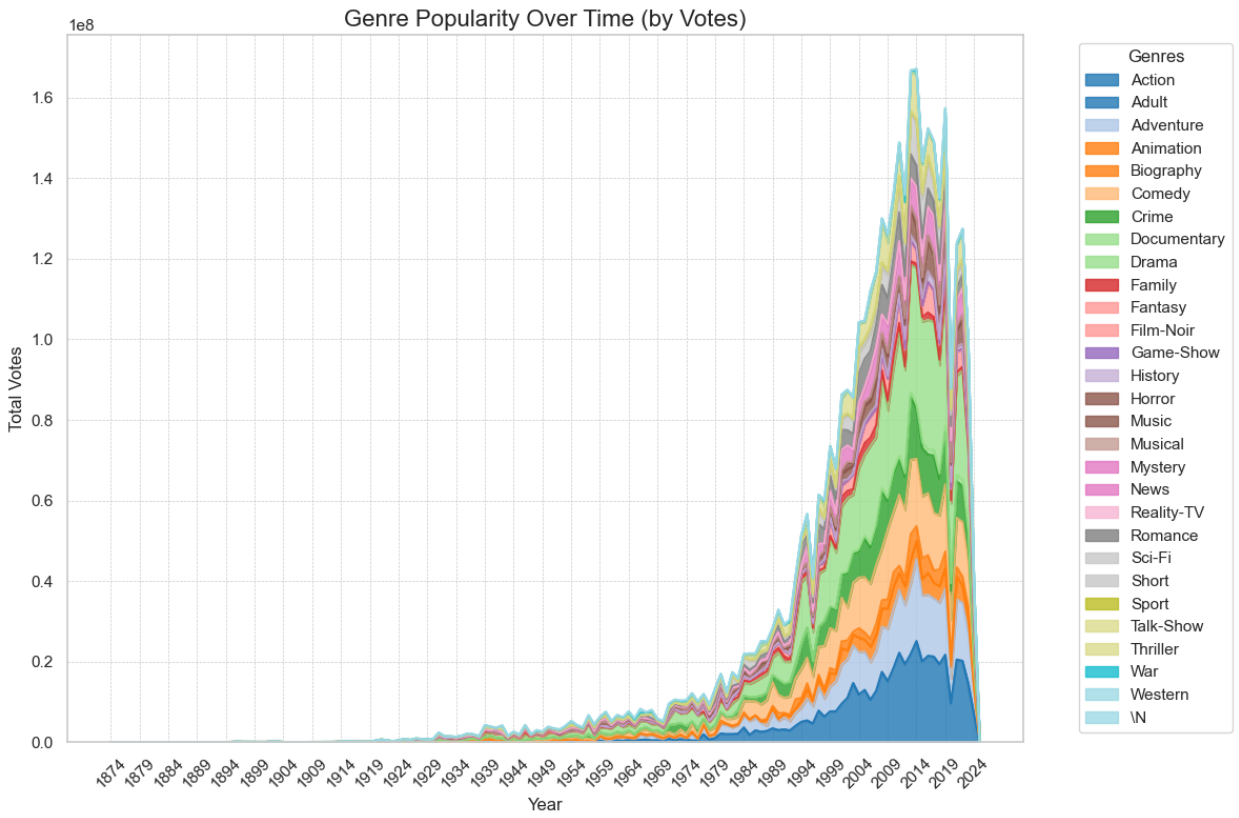
Objective:

To explore how the popularity of different movie genres has evolved over time, as measured by user votes.

In [16]:

```
# Separate multiple genres into individual rows
df_title_basics['genres'] = df_title_basics['genres'].str.split(',')
df_title_basics_exploded = df_title_basics.explode('genres')

genre_popularity = df_title_basics_exploded.merge(df_ratings, on='tcon

genre_popularity['startYear'] = pd.to_numeric(genre_popularity['startY
genre_popularity = genre_popularity.dropna(subset=['startYear'])
genre_popularity['startYear'] = genre_popularity['startYear'].astype(i
genre_popularity = genre_popularity.groupby(['startYear', 'genres'])['

plt.figure(figsize=(12, 8))
genre_popularity.plot(kind='area', stacked=True, figsize=(12, 8), colo
plt.title('Genre Popularity Over Time (by Votes)', fontsize=16)
plt.xlabel('Year', fontsize=12)
plt.ylabel('Total Votes', fontsize=12)
plt.xticks(ticks=range(genre_popularity.index.min(), genre_popularity.
plt.legend(title="Genres", bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.tight_layout()
plt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```



Results:

The graph "Genre Popularity Over Time (by Votes)" shows that action, drama, and comedy have consistently remained the most popular genres by votes. In recent years, there has been a significant rise in sci-fi and superhero movies, reflecting the growing trend of fantasy and big-budget franchises.

Interpretation: Genre preferences have shifted over time, with action movies dominating in recent decades, aligning with the growth of blockbuster franchises and global cinematic appeal.

3.8 Lead Actor Age and Success

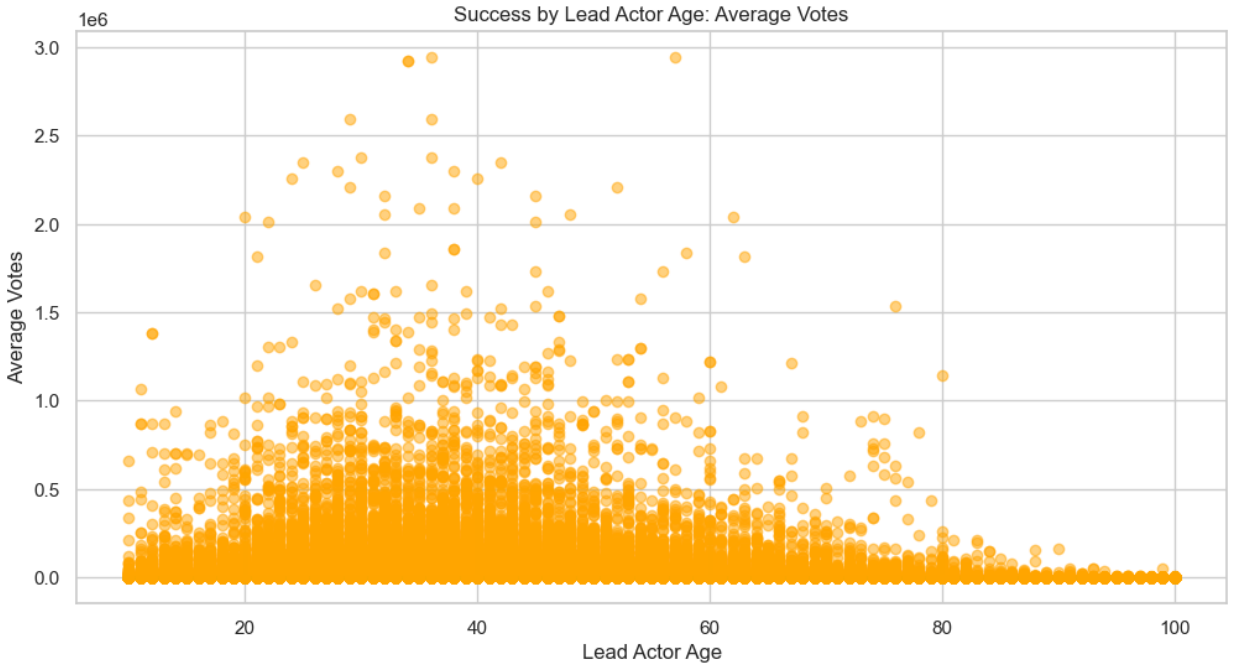
Objective:

To investigate the relationship between the lead actor's age and movie success, measured by both average ratings and votes. This analysis aims to explore whether specific age groups of lead actors are correlated with higher movie ratings or increased user engagement (votes).

In [17]:
```
df_movies = df_title_basics.merge(df_ratings, on='tconst')

df_movies['startYear'] = pd.to_numeric(df_movies['startYear'], errors=
df_movies = df_movies.dropna(subset=['startYear', 'averageRating', 'nu
df_movies['startYear'] = df_movies['startYear'].astype(int)
```

In [18]:
```

```

```
# Get only lead actors
df_principals['role'] = df_principals['ordering'].apply(lambda x: 'Lea

df_lead_actors = df_principals[df_principals['role'] == 'Lead'].merge(
df_lead_actors['birthYear'] = pd.to_numeric(df_lead_actors['birthYear'
df_lead_actors = df_lead_actors.dropna(subset=['birthYear'])
df_movies_lead_actors = df_movies.merge(df_lead_actors[['tconst', 'nco
df_movies_lead_actors['actor_age'] = df_movies_lead_actors['startYear'
df_movies_lead_actors_filtered = df_movies_lead_actors[df_movies_lead_

plt.figure(figsize=(12, 6))
plt.scatter(df_movies_lead_actors_filtered['actor_age'], df_movies_lea
plt.xlabel('Lead Actor Age')
plt.ylabel('Average Votes')
plt.title('Success by Lead Actor Age: Average Votes')
plt.grid(True)
plt.show()
```



Results:

The following graphs depict how lead actor age correlates with both average movie ratings and the number of votes. The scatterplot "Success by Lead Actor Age: Average Votes" shows a higher concentration of user votes for movies with younger lead actors, while average ratings are relatively stable across various actor ages.

- **Average Ratings:** There seems to be no strong linear correlation between the lead actor's age and the average rating of movies, with stable ratings for various age groups.
- **Average Votes:** Movies with younger lead actors (20s to 40s) tend to receive significantly more votes. As actor age increases beyond 40, the number of votes generally decreases.

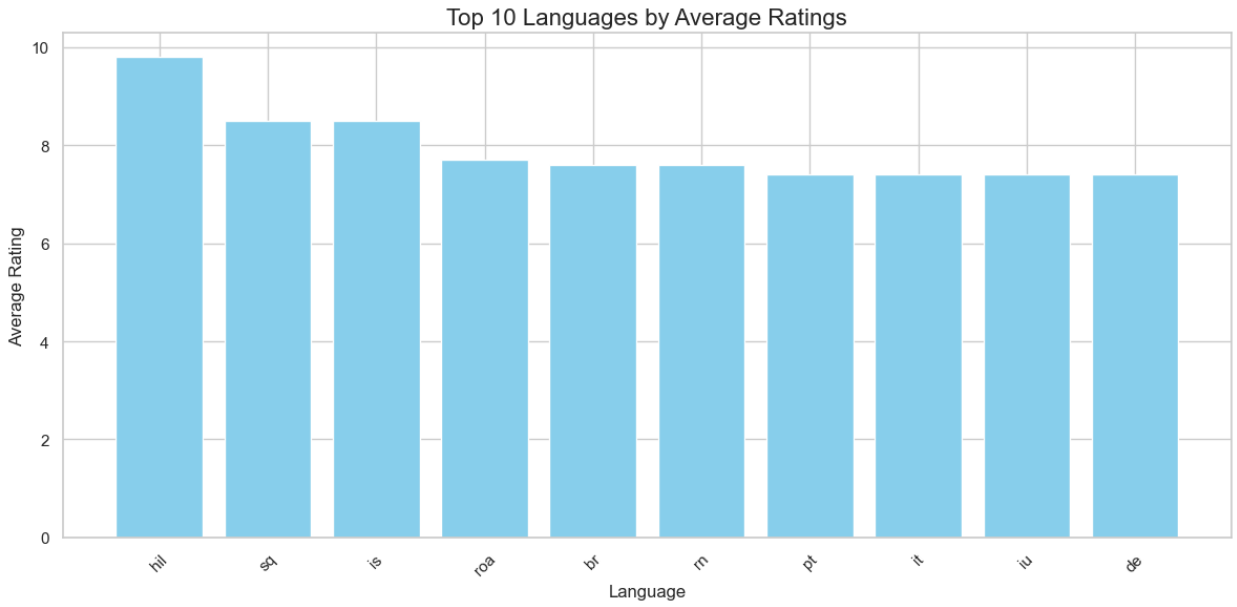
3.9 Success by Language

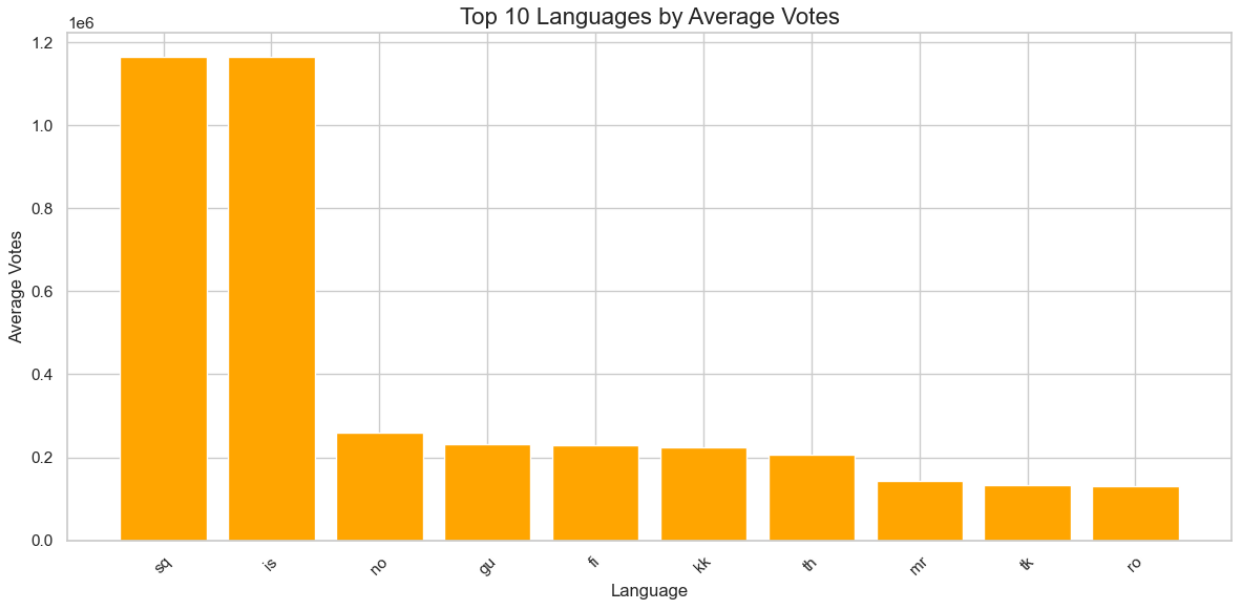
Objective:

To examine how the primary language of a movie impacts its success, measured by average ratings and votes. This analysis helps determine if movies in specific languages are rated higher or receive more votes.

In [19]:

```
df_movies_akas = df_akas.merge(df_ratings, left_on='titleId', right_or

# Group by language and calculate the average rating and votes
language_success = df_movies_akas.groupby('language').agg({
    'averageRating': 'mean',
    'numVotes': 'mean'
}).reset_index()


language_success_filtered = language_success[language_success['languag
top_languages_by_rating = language_success_filtered.sort_values(by='av

plt.figure(figsize=(12, 6))
plt.bar(top_languages_by_rating['language'], top_languages_by_rating['
plt.xlabel('Language', fontsize=12)
plt.ylabel('Average Rating', fontsize=12)
plt.title('Top 10 Languages by Average Ratings', fontsize=16)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

top_languages_by_votes = language_success_filtered.sort_values(by='num

plt.figure(figsize=(12, 6))
plt.bar(top_languages_by_votes['language'], top_languages_by_votes['nu
plt.xlabel('Language', fontsize=12)
plt.ylabel('Average Votes', fontsize=12)
plt.title('Top 10 Languages by Average Votes', fontsize=16)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```





Results:

The bar graphs below display the top 10 languages by average rating and average votes. The languages 'hil' (possibly Hiligaynon), 'sq' (Albanian), and 'is' (Icelandic) emerge as the highest rated. However, the top votes tend to be concentrated among other languages such as 'sq' and 'is'.

- **Average Ratings:** Movies in certain languages (e.g., Hiligaynon and Albanian) receive high average ratings, though this could be attributed to a smaller sample size. Further analysis is required to confirm this.
- **Average Votes:** Languages like Albanian ('sq') and Icelandic ('is') attract a significantly larger number of votes, suggesting higher user engagement for movies in these languages.

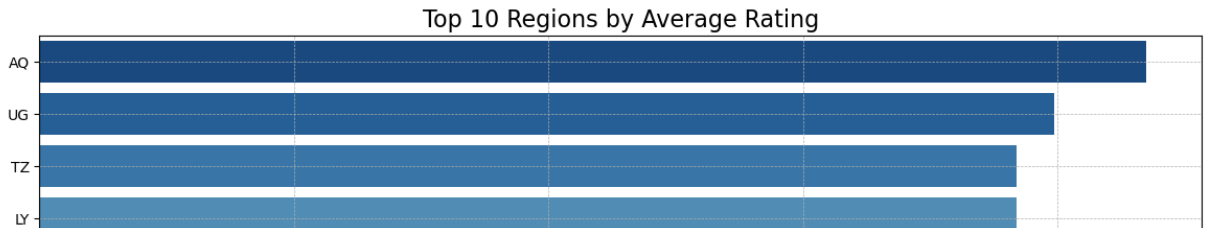
3.10 Success by Region

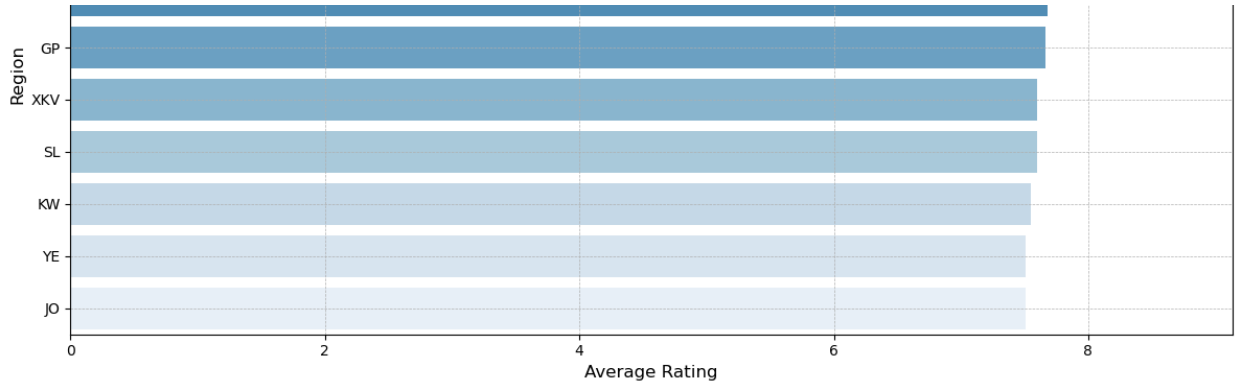
Objective:

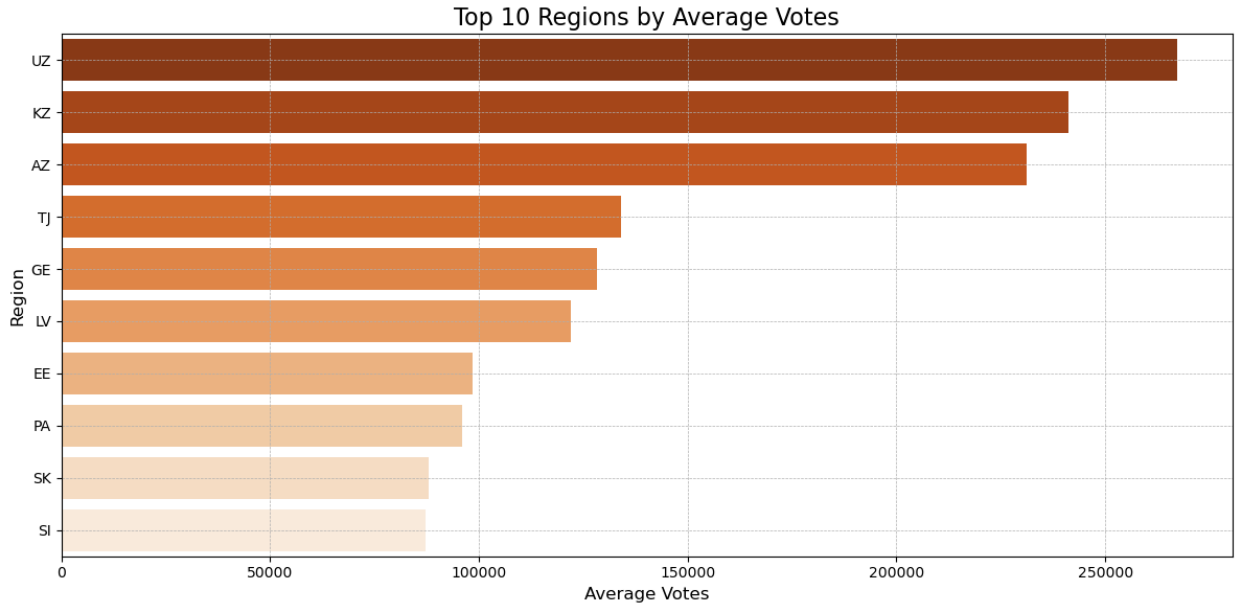
To explore how the region associated with a movie influences its success, as measured by average ratings and votes. This helps identify whether certain geographic regions tend to produce more successful films.

In [9]:

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

# Merge akas with ratings to include rating and votes data
df_movies_akas_region = df_akas.merge(df_ratings, left_on='titleId', r

# Group by region and calculate the average rating and votes
region_success = df_movies_akas_region.groupby('region').agg({
    'averageRating': 'mean',
    'numVotes': 'mean'
}).reset_index()

# Filter out rows where region is NaN or empty
region_success = region_success[region_success['region'].notnull() & (

# Sort by average rating and get top 10 regions by average rating
top_10_regions_rating = region_success.sort_values(by='averageRating',

# Sort by average votes and get top 10 regions by average votes
top_10_regions_votes = region_success.sort_values(by='numVotes', ascen

# Plot top 10 regions by average rating (Horizontal Bar Graph)
plt.figure(figsize=(12, 6))
sns.barplot(x='averageRating', y='region', data=top_10_regions_rating,
plt.title('Top 10 Regions by Average Rating', fontsize=16)
plt.xlabel('Average Rating', fontsize=12)
plt.ylabel('Region', fontsize=12)
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.tight_layout()
plt.show()

# Plot top 10 regions by average votes (Horizontal Bar Graph)
plt.figure(figsize=(12, 6))
sns.barplot(x='numVotes', y='region', data=top_10_regions_votes, palet
plt.title('Top 10 Regions by Average Votes', fontsize=16)
plt.xlabel('Average Votes', fontsize=12)
plt.ylabel('Region', fontsize=12)
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.tight_layout()
plt.show()
```







Results:

The bar charts below illustrate the top 10 regions by average rating and average votes. Movies from 'AQ' (Antarctica) and 'UG' (Uganda) top the ratings chart, while regions like 'UZ' (Uzbekistan) and 'KZ' (Kazakhstan) dominate the votes chart.

- **Average Ratings:** Some regions like 'Antarctica' and 'Uganda' stand out with exceptionally high average ratings, although these regions may have fewer movies contributing to the data.
- **Average Votes:** Regions such as Uzbekistan and Kazakhstan receive the highest number of votes, indicating higher engagement from audiences in these areas.

4. Defining a Hit Movie Metric

The goal is to define a metric that captures the notion of a 'hit movie' and provide recommendations to a movie studio director looking to invest in building the next AAA title. This metric is based on both the **average rating** and **user votes** from the dataset. The aim is to balance **critical success** (measured by ratings) with **audience engagement** (measured by votes).

The analysis has revealed various factors that impact a movie's rating and audience engagement, such as **lead actor age**, **movie runtime**, **language**, **region**, and **genre popularity**. Based on these results, we propose a metric that combines average ratings and total votes to assess movie success.

The formula for this hit movie metric is as follows:

```
Hit Movie Score = (α × Normalized Average Rating) + (β × Norma
lized Total Votes)
```

Where:

- α and β are weights that can be adjusted based on whether the studio prioritizes audience engagement (votes) or critical success (ratings).
- Normalized values scale the ratings and votes to comparable ranges.

Based on the results from the previous analysis, we can provide the following recommendations to optimize both ratings and votes for the next AAA title:

A. Lead Actor Age and Success

Younger lead actors (aged 20-40) tend to attract more votes. The analysis shows that this age group generally garners higher audience engagement, while ratings across different age groups remain relatively stable.

Recommendation:

Cast lead actors aged between 20-40 years to maximize audience engagement and increase the likelihood of higher votes.

B. Movie Runtime and Success

Movies with a runtime of 90-120 minutes consistently receive higher ratings. Longer movies (>120 minutes) tend to see a decrease in both ratings and votes, while shorter movies (<90 minutes) also perform well in some cases.

Recommendation:

Aim for a movie runtime between 90-120 minutes to balance viewer satisfaction and maintain engagement. For specific genres like comedy or animated films, a shorter runtime can also work effectively.

C. Success by Language

Languages such as English and Spanish show high engagement in terms of votes, while some lesser-used languages like Hiligaynon and Albanian exhibit high average ratings, although likely due to smaller datasets.

Recommendation:

Use English or Spanish as the primary language for broad appeal and higher audience engagement. Consider additional dubbing or subtitles to cater to specific regional or niche markets.

D. Success by Region

Movies from regions such as Uzbekistan and Kazakhstan receive a high number of votes, reflecting strong engagement from audiences in these regions. Regions like Antarctica and Uganda score high average ratings, but likely with smaller sample sizes.

Recommendation:

Focus promotional efforts in regions with strong voting trends, like Uzbekistan and Kazakhstan, while ensuring broader global appeal to maximize reach.

E. Genre Popularity

Genres like action, drama, and comedy consistently show high engagement, as measured by votes. Sci-fi and superhero genres have also seen a significant rise in popularity in recent years.

Recommendation:

Focus on action, drama, and comedy for broader appeal. For emerging trends, consider investing in sci-fi or superhero genres to tap into growing audience preferences.

5. Conclusion

In conclusion, the analysis provides several key recommendations for producing a hit movie. First, casting younger lead actors, particularly those aged between 20-40, is likely to increase audience engagement and generate more votes. Additionally, opting for a movie runtime between 90 and 120 minutes ensures high viewer satisfaction while keeping the audience engaged. This runtime strikes a balance between delivering sufficient plot development and maintaining viewer interest.

Language is another important factor to consider. Using English or Spanish as the primary language maximizes global appeal, as these languages have proven to attract higher audience engagement. For regional or niche markets, dubbing or providing subtitles in additional languages can further broaden the movie's reach.

In terms of regional success, targeting regions like Uzbekistan and Kazakhstan for promotional efforts is a smart move, as these regions demonstrate strong voting trends. However, it is also important to ensure global reach by focusing on larger, more diverse markets.

Finally, when choosing a genre, it is beneficial to focus on popular genres like action, drama, and comedy, as they consistently receive high engagement. However, for studios looking to tap into emerging trends, the sci-fi and superhero genres are also gaining traction and could attract growing audiences.

By considering these factors—lead actor age, movie runtime, language, regional focus, and genre—a studio director can create a balanced strategy that maximizes both critical success and audience engagement, leading to the production of a hit movie.