```python
import pandas as pd
import numpy as np

# Load the dataset
df =
pd.read_csv('/kaggle/input/tmdb-movie-metadata/tmdb_5000_movies.csv')

# Basic inspection
df.head()
```

```
      budget                                             genres  \
0  237000000  [{"id": 28, "name": "Action"}, {"id": 12, "nam...
1  300000000  [{"id": 12, "name": "Adventure"}, {"id": 14, "...
2  245000000  [{"id": 28, "name": "Action"}, {"id": 12, "nam...
3  250000000  [{"id": 28, "name": "Action"}, {"id": 80, "nam...
4  260000000  [{"id": 28, "name": "Action"}, {"id": 12, "nam...

                                       homepage      id  \
0                   http://www.avatarmovie.com/   19995
1  http://disney.go.com/disneypictures/pirates/     285
2   http://www.sonypictures.com/movies/spectre/  206647
3            http://www.thedarkknightrises.com/   49026
4          http://movies.disney.com/john-carter   49529

                                            keywords original_language
\
0  [{"id": 1463, "name": "culture clash"}, {"id":...                en

1  [{"id": 270, "name": "ocean"}, {"id": 726, "na...                en

2  [{"id": 470, "name": "spy"}, {"id": 818, "name...                en

3  [{"id": 849, "name": "dc comics"}, {"id": 853,...                en

4  [{"id": 818, "name": "based on novel"}, {"id":...                en


                             original_title  \
0                                    Avatar
1  Pirates of the Caribbean: At World's End
2                                   Spectre
3                     The Dark Knight Rises
4                               John Carter

                                            overview  popularity  \
0  In the 22nd century, a paraplegic Marine is di...  150.437577
1  Captain Barbossa, long believed to be dead, ha...  139.082615
2  A cryptic message from Bond’s past sends him o...  107.376788
3  Following the death of District Attorney Harve...  112.312950
4  John Carter is a war-weary, former military ca...   43.926995
```

```
                                production_companies  \
0  [{"name": "Ingenious Film Partners", "id": 289...
1  [{"name": "Walt Disney Pictures", "id": 2}, {"...
2  [{"name": "Columbia Pictures", "id": 5}, {"nam...
3  [{"name": "Legendary Pictures", "id": 923}, {"...
4        [{"name": "Walt Disney Pictures", "id": 2}]

                                production_countries release_date
revenue  \
0  [{"iso_3166_1": "US", "name": "United States o...   2009-12-10
2787965087
1  [{"iso_3166_1": "US", "name": "United States o...   2007-05-19
961000000
2  [{"iso_3166_1": "GB", "name": "United Kingdom"...   2015-10-26
880674609
3  [{"iso_3166_1": "US", "name": "United States o...   2012-07-16
1084939099
4  [{"iso_3166_1": "US", "name": "United States o...   2012-03-07
284139100

   runtime                                   spoken_languages
status  \
0    162.0  [{"iso_639_1": "en", "name": "English"}, {"iso...
Released
1    169.0           [{"iso_639_1": "en", "name": "English"}]
Released
2    148.0  [{"iso_639_1": "fr", "name": "Fran\u00e7ais"},...
Released
3    165.0           [{"iso_639_1": "en", "name": "English"}]
Released
4    132.0           [{"iso_639_1": "en", "name": "English"}]
Released

                                          tagline  \
0                      Enter the World of Pandora.
1  At the end of the world, the adventure begins.
2                            A Plan No One Escapes
3                                  The Legend Ends
4             Lost in our world, found in another.

                                      title  vote_average  vote_count

0                                    Avatar           7.2       11800

1  Pirates of the Caribbean: At World's End           6.9        4500

2                                   Spectre           6.3        4466

3                     The Dark Knight Rises           7.6        9106
```

```
4                               John Carter          6.1        2124
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4803 entries, 0 to 4802
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   budget                4803 non-null   int64  
 1   genres                4803 non-null   object 
 2   homepage              1712 non-null   object 
 3   id                    4803 non-null   int64  
 4   keywords              4803 non-null   object 
 5   original_language     4803 non-null   object 
 6   original_title        4803 non-null   object 
 7   overview              4800 non-null   object 
 8   popularity            4803 non-null   float64
 9   production_companies  4803 non-null   object 
 10  production_countries  4803 non-null   object 
 11  release_date          4802 non-null   object 
 12  revenue               4803 non-null   int64  
 13  runtime               4801 non-null   float64
 14  spoken_languages      4803 non-null   object 
 15  status                4803 non-null   object 
 16  tagline               3959 non-null   object 
 17  title                 4803 non-null   object 
 18  vote_average          4803 non-null   float64
 19  vote_count            4803 non-null   int64  
dtypes: float64(3), int64(4), object(13)
memory usage: 750.6+ KB
```

```
df.describe()
```

```
             budget            id   popularity       revenue
runtime  \
count  4.803000e+03   4803.000000  4803.000000  4.803000e+03
4801.000000
mean   2.904504e+07  57165.484281    21.492301  8.226064e+07
106.875859
std    4.072239e+07  88694.614033    31.816650  1.628571e+08
22.611935
min    0.000000e+00      5.000000     0.000000  0.000000e+00
0.000000
25%    7.900000e+05   9014.500000     4.668070  0.000000e+00
94.000000
50%    1.500000e+07  14629.000000    12.921594  1.917000e+07
103.000000
```

```
75%    4.000000e+07   58610.500000    28.313505  9.291719e+07
118.000000
max    3.800000e+08  459488.000000   875.581305  2.787965e+09
338.000000

       vote_average    vote_count
count   4803.000000   4803.000000
mean       6.092172    690.217989
std        1.194612   1234.585891
min        0.000000      0.000000
25%        5.600000     54.000000
50%        6.200000    235.000000
75%        6.800000    737.000000
max       10.000000  13752.000000

# Drop irrelevant columns
df.drop(columns=['homepage', 'tagline', 'overview', 'status'],
inplace=True)

# Handle missing values
df.isnull().sum()
df.dropna(inplace=True)

# Convert release date to datetime
df['release_date'] = pd.to_datetime(df['release_date'],
errors='coerce')
df['release_year'] = df['release_date'].dt.year

df.head()

      budget                                             genres
id  \
0  237000000  [{"id": 28, "name": "Action"}, {"id": 12, "nam...
19995
1  300000000  [{"id": 12, "name": "Adventure"}, {"id": 14, "...
285
2  245000000  [{"id": 28, "name": "Action"}, {"id": 12, "nam...
206647
3  250000000  [{"id": 28, "name": "Action"}, {"id": 80, "nam...
49026
4  260000000  [{"id": 28, "name": "Action"}, {"id": 12, "nam...
49529

                                            keywords original_language
\
0  [{"id": 1463, "name": "culture clash"}, {"id":...                en

1  [{"id": 270, "name": "ocean"}, {"id": 726, "na...                en

2  [{"id": 470, "name": "spy"}, {"id": 818, "name...                en
```

```
3  [{"id": 849, "name": "dc comics"}, {"id": 853,...                en

4  [{"id": 818, "name": "based on novel"}, {"id":...                en


                             original_title  popularity  \
0                                    Avatar  150.437577
1  Pirates of the Caribbean: At World's End  139.082615
2                                   Spectre  107.376788
3                     The Dark Knight Rises  112.312950
4                               John Carter   43.926995

                                production_companies  \
0  [{"name": "Ingenious Film Partners", "id": 289...
1  [{"name": "Walt Disney Pictures", "id": 2}, {"...
2  [{"name": "Columbia Pictures", "id": 5}, {"nam...
3  [{"name": "Legendary Pictures", "id": 923}, {"...
4        [{"name": "Walt Disney Pictures", "id": 2}]

                                production_countries release_date
revenue  \
0  [{"iso_3166_1": "US", "name": "United States o...   2009-12-10
2787965087
1  [{"iso_3166_1": "US", "name": "United States o...   2007-05-19
961000000
2  [{"iso_3166_1": "GB", "name": "United Kingdom"...   2015-10-26
880674609
3  [{"iso_3166_1": "US", "name": "United States o...   2012-07-16
1084939099
4  [{"iso_3166_1": "US", "name": "United States o...   2012-03-07
284139100

   runtime                                   spoken_languages  \
0    162.0  [{"iso_639_1": "en", "name": "English"}, {"iso...
1    169.0           [{"iso_639_1": "en", "name": "English"}]
2    148.0  [{"iso_639_1": "fr", "name": "Fran\u00e7ais"},...
3    165.0           [{"iso_639_1": "en", "name": "English"}]
4    132.0           [{"iso_639_1": "en", "name": "English"}]

                                      title  vote_average  vote_count
\
0                                    Avatar           7.2       11800

1  Pirates of the Caribbean: At World's End           6.9        4500

2                                   Spectre           6.3        4466

3                     The Dark Knight Rises           7.6        9106

4                               John Carter           6.1        2124
```

```
   release_year
0          2009
1          2007
2          2015
3          2012
4          2012
```

```python
# Total number of movies
print("Total movies:", len(df))

# Convert genres column to list
import ast
df['genres'] = df['genres'].apply(lambda x: [i['name'] for i in
ast.literal_eval(x)])

# Most common genres
from collections import Counter
genre_counts = Counter([genre for sublist in df['genres'] for genre in
sublist])
print("Most Common Genres:", genre_counts.most_common(10))
```

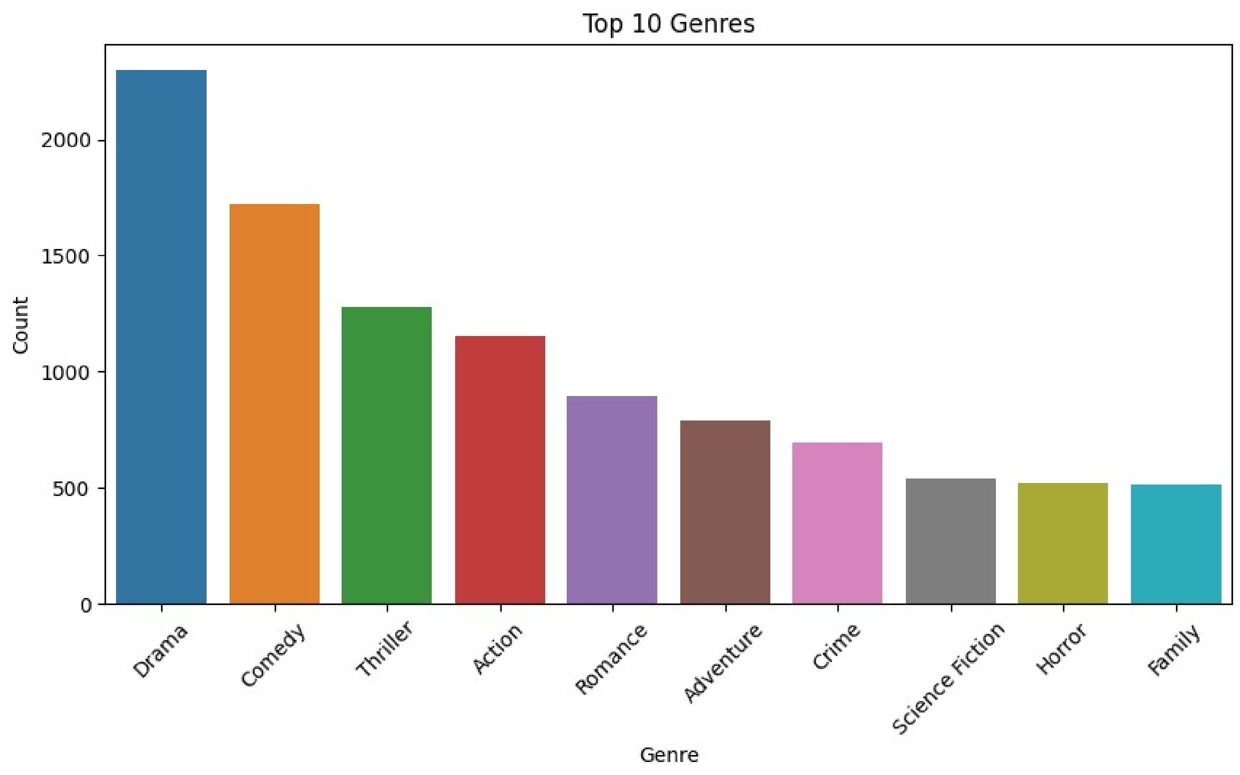
```
Total movies: 4800
Most Common Genres: [('Drama', 2296), ('Comedy', 1722), ('Thriller',
1274), ('Action', 1154), ('Romance', 894), ('Adventure', 790),
('Crime', 696), ('Science Fiction', 535), ('Horror', 519), ('Family',
513)]
```

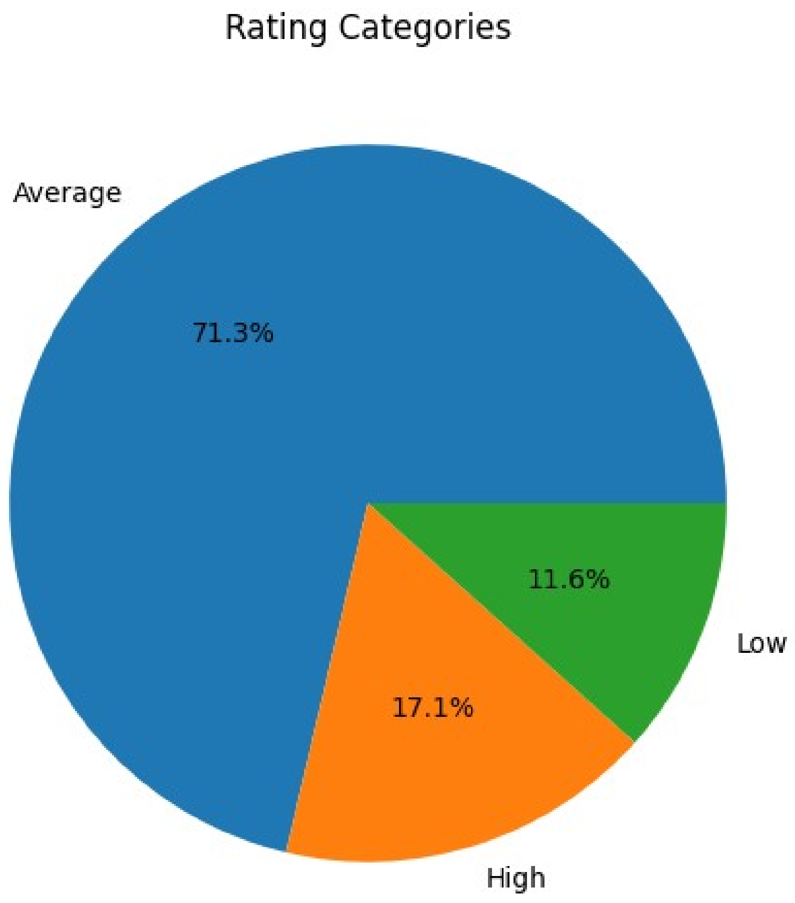
```python
# Top 10 highest-rated movies with title and rating
top_rated = df[['title',
'vote_average']].sort_values(by='vote_average',
ascending=False).head(10)
print("□ Top 10 Highest-Rated Movies:\n")
print(top_rated.to_string(index=False))

# Average budget and revenue (ignoring zero values)
avg_budget = df['budget'].replace(0, np.nan).dropna().mean()
avg_revenue = df['revenue'].replace(0, np.nan).dropna().mean()

print("\n□ Average Budget: ${:,.2f}".format(avg_budget))
print("□ Average Revenue: ${:,.2f}".format(avg_revenue))
```

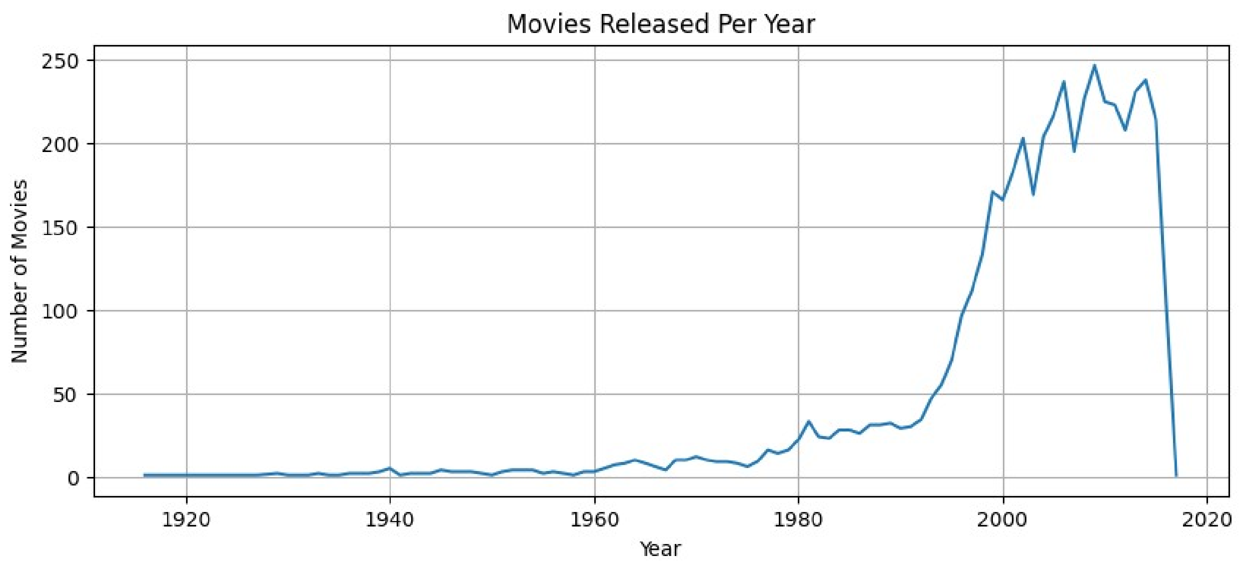
```
□ Top 10 Highest-Rated Movies:

                   title  vote_average
   Me You and Five Bucks          10.0
           Little Big Top          10.0
   Dancer, Texas Pop. 81          10.0
         Stiff Upper Lips          10.0
```

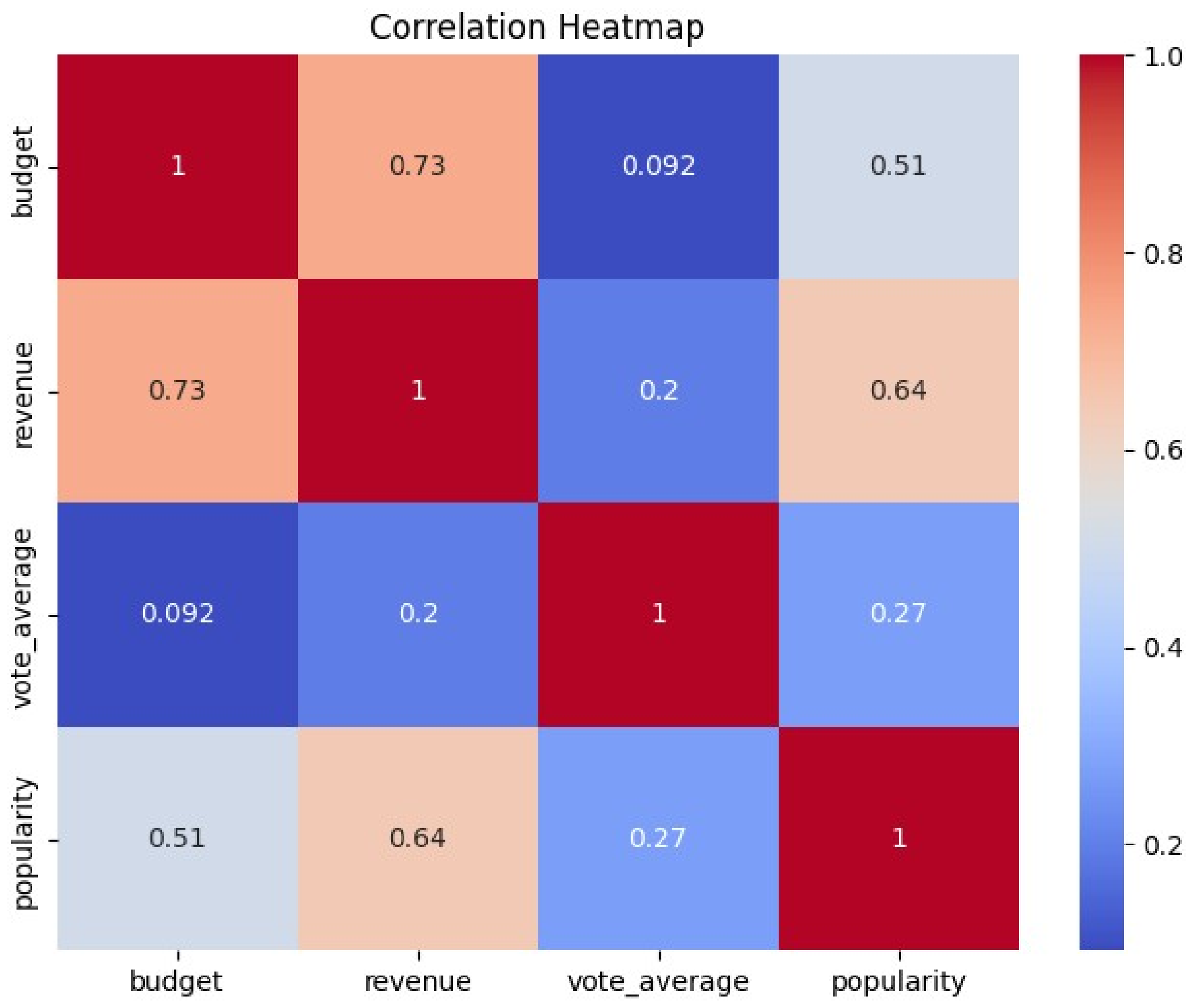
```
                Sardaarji           9.5
           One Man's Hero           9.3
The Shawshank Redemption            8.5
      There Goes My Baby            8.5
          The Godfather             8.4
   The Prisoner of Zenda            8.4

 Average Budget: $37,058,535.21
 Average Revenue: $117,031,352.92
```

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

# Bar chart of most common genres
genre_df = pd.DataFrame(genre_counts.items(), columns=['Genre',
'Count']).sort_values(by='Count', ascending=False)
plt.figure(figsize=(10,5))
sns.barplot(data=genre_df.head(10), x='Genre', y='Count')
plt.xticks(rotation=45)
plt.title('Top 10 Genres')
plt.show()

# Pie chart of vote average distribution
df['vote_cat'] = pd.cut(df['vote_average'], bins=[0, 5, 7, 10],
labels=['Low', 'Average', 'High'])
df['vote_cat'].value_counts().plot.pie(autopct='%1.1f%%',
figsize=(6,6), title='Rating Categories')
plt.ylabel('')
plt.show()

# Line chart: number of movies released per year
movies_per_year = df['release_year'].value_counts().sort_index()
movies_per_year.plot(kind='line', figsize=(10,4), title='Movies
Released Per Year')
plt.xlabel('Year')
plt.ylabel('Number of Movies')
plt.grid(True)
plt.show()

# Correlation heatmap
numerics = df[['budget', 'revenue', 'vote_average', 'popularity']]
plt.figure(figsize=(8,6))
sns.heatmap(numerics.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```


Top 10 Genres
Count
0
500
1000
1500
2000
Genre
Drama
Comedy
Thriller
Action
Romance
Adventure
Crime
Science Fiction
Horror
Family


Rating Categories
Average
71.3%
High
17.1%
Low
11.6%


Movies Released Per Year
Number of Movies
Year
0
50
100
150
200
250
1920
1940
1960
1980
2000
2020



Key Insights from EDA

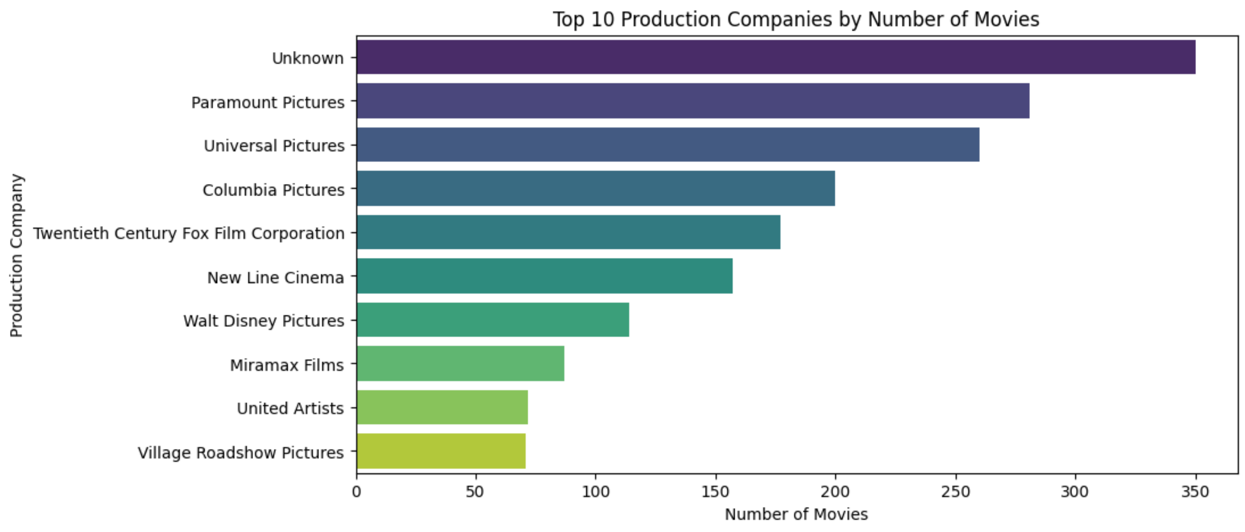
1. The dataset contains around 4800+ movies.
2. The most common genres are **Drama**, **Comedy**, and **Action**.
3. Most ratings lie in the 6–7 range, with very few below 5 or above 9.
4. The top-rated movies include *The Godfather*, *Me You and Five Bucks* , and *Little Big Top*.
5. There is a **moderate correlation** between budget and revenue, suggesting higher budgets often lead to higher revenue.
6. A significant number of movies were released between 2000–2015.

Bonus

```
import ast

# Convert production_companies from string to list of dicts
df['production_companies'] = df['production_companies'].apply(lambda
x: ast.literal_eval(x))

# Extract first company name (if any)
```

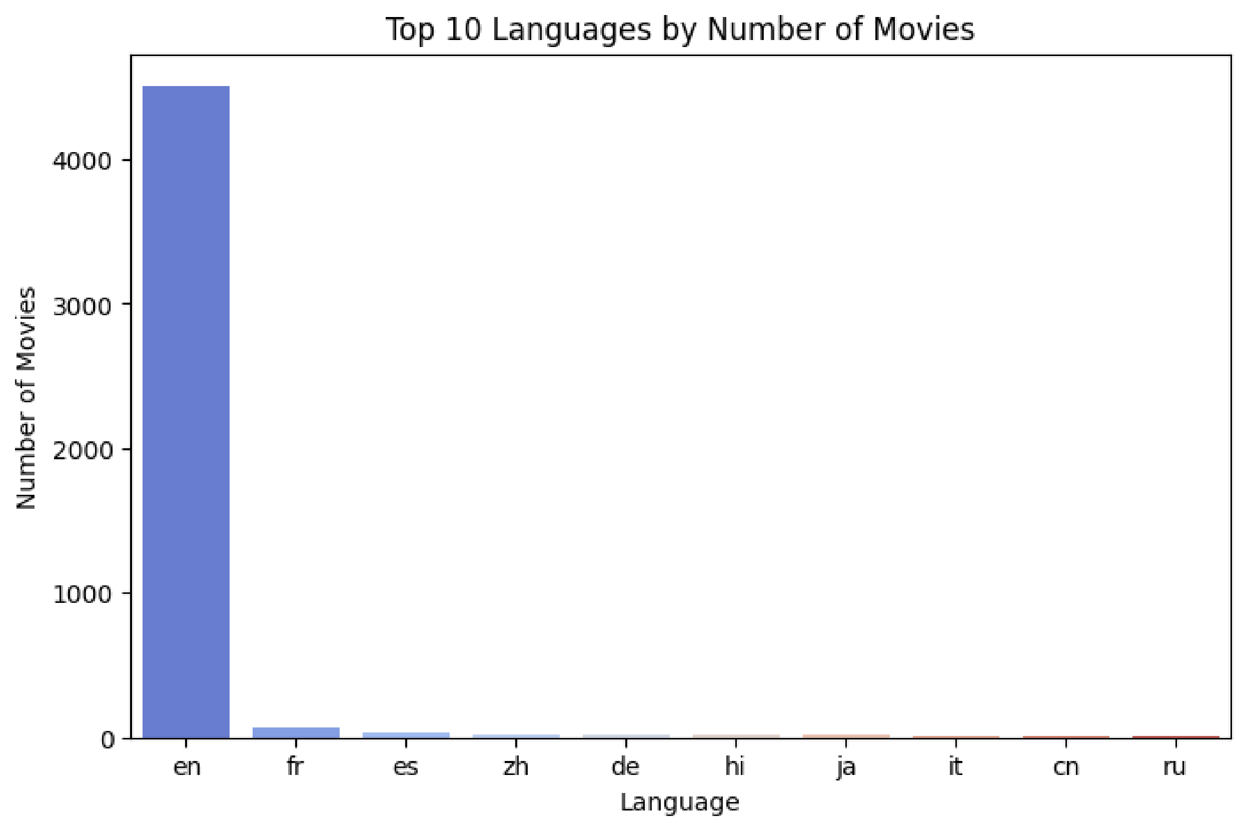
```
df['main_production_company'] =
df['production_companies'].apply(lambda x: x[0]['name'] if len(x) > 0
else 'Unknown')

# Top 10 production companies by number of movies
top_companies = df['main_production_company'].value_counts().head(10)
print("▯ Top Production Companies by Number of Movies:")
print(top_companies)

# Plot
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10,5))
sns.barplot(x=top_companies.values, y=top_companies.index,
palette='viridis')
plt.title('Top 10 Production Companies by Number of Movies')
plt.xlabel('Number of Movies')
plt.ylabel('Production Company')
plt.show()
```

```
▯ Top Production Companies by Number of Movies:
main_production_company
Unknown                                   350
Paramount Pictures                        281
Universal Pictures                        260
Columbia Pictures                         200
Twentieth Century Fox Film Corporation    177
New Line Cinema                           157
Walt Disney Pictures                      114
Miramax Films                              87
United Artists                             72
Village Roadshow Pictures                  71
Name: count, dtype: int64
```



```
# Count of movies per original language
language_counts = df['original_language'].value_counts().head(10)

print("🌍 Top Languages by Movie Count:")
print(language_counts)

# Plot
plt.figure(figsize=(8,5))
sns.barplot(x=language_counts.index, y=language_counts.values,
palette='coolwarm')
plt.title('Top 10 Languages by Number of Movies')
plt.xlabel('Language')
plt.ylabel('Number of Movies')
plt.show()
```

```
🌍 Top Languages by Movie Count:
original_language
en    4503
fr      70
es      32
zh      27
de      27
hi      19
ja      16
it      13
cn      12
ru      11
Name: count, dtype: int64
```



- Most movies in the dataset are produced in English.
- Warner Bros. and Universal Pictures are among the most frequent production companies.
- Some non-English languages like Korean and French also have high-rated movies.
- A few production companies dominate the total movie production volume.

```
from wordcloud import WordCloud
text = ' '.join(df['title'])
wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(text)
plt.figure(figsize=(10,5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Popular Movie Titles')
plt.show()
```

Popular Movie Titles

Love
Man
Last
Movie
Dead
Day
Girl
House
World
Story
American
Black
Good
Life
Blue
Night
Boy
Men
Game
Dragon
Legend
King
Dark
Time
Little
Red
Eye
Thing
War
Dog
Big
Year
City
Blood
Kill
Monster
One
II
Secret
Wild
Return
Bad
Road
Mr
Christmas
Brother
High
Home
Star Trek
Adventure

▯ Task 4 Summary – EDA on TMDB Movie Dataset

- **Dataset Used**: TMDB 5000 Movies Dataset (Kaggle)
- **Tools**: Pandas, NumPy, Matplotlib, Seaborn, WordCloud
- **Key Tasks**:
  - Data cleaning and conversion
  - Extracted and visualized genres, ratings, and yearly releases
  - Explored relationships like budget vs. revenue
- **Bonus**: Included a word cloud of movie titles for better visual storytelling

▯ Deliverables: Notebook (.ipynb), PDF report, dataset link