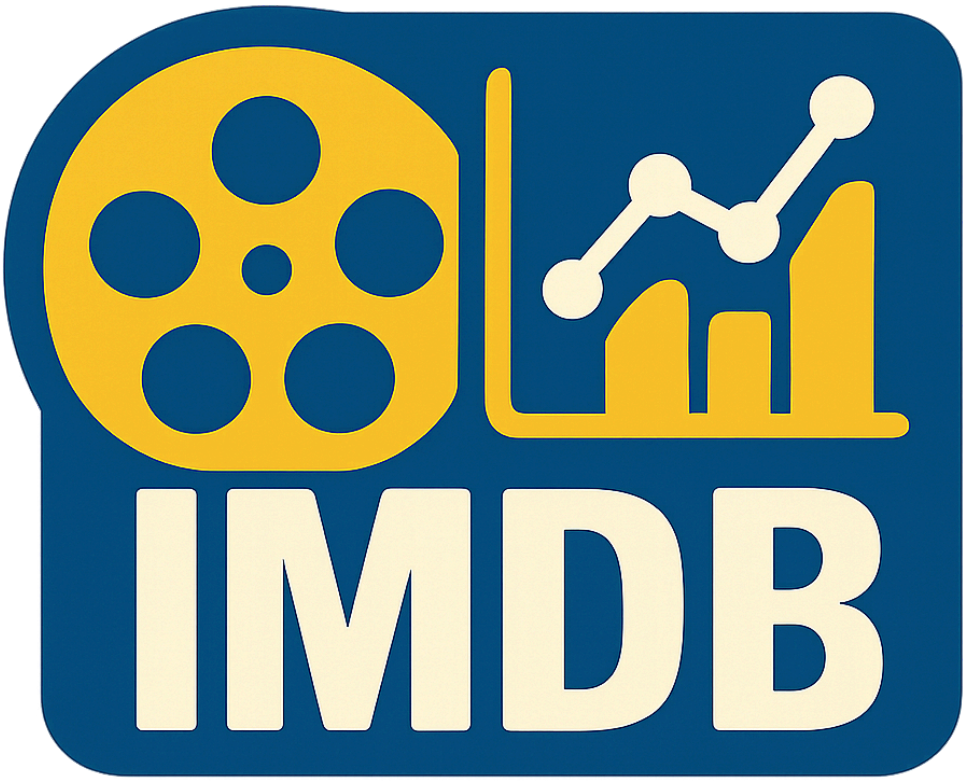


About the Dataset

Here's a data set of 1,000 most popular movies on IMDB in the last 10 years. The data fields included are:

Title, Genre, Description, Director, Actors, Year, Runtime, Rating, Votes, Revenue, Metascrore

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv("IMDB-Movie-Data.csv")
df.head()
```

Out[2]:

	Rank	Title	Genre	Description	Director	Acto
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, V Diesel, Brad Cooper, Zoe
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapa Logan Marsha Green, Michael F
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAv Anya Taylor-J Haley Lu Richa
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matth McConaughey,Ree Witherspoon, Se M
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jar Leto, Marg Robbie, Viola I

In [3]:
```
df.tail()
```

Out[3]:

	Rank	Title	Genre	Description	Director	Actors	Yea
995	996	Secret in Their Eyes	Crime,Drama,Mystery	A tight-knit team of rising investigators, alo...	Billy Ray	Chiwetel Ejiofor, Nicole Kidman, Julia Roberts...	201
996	997	Hostel: Part II	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...	200
997	998	Step Up 2: The Streets	Drama,Music,Romance	Romantic sparks occur between two dance studen...	Jon M. Chu	Robert Hoffman, Briana Evigan, Cassie Ventura,...	200
998	999	Search Party	Adventure,Comedy	A pair of friends embark on a mission to reuni...	Scot Armstrong	Adam Pally, T.J. Miller, Thomas Middleditch,Sh...	201
999	1000	Nine Lives	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	201

In [4]:
```
df.shape
```

Out[4]:
```
(1000, 12)
```

In [5]:
```
df.shape[0]
```

Out[5]:
```
1000
```

In [6]:
```
df.shape[1]
```

Out[6]:
```
12
```

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Rank                1000 non-null   int64  
 1   Title               1000 non-null   object 
 2   Genre               1000 non-null   object 
 3   Description         1000 non-null   object 
 4   Director            1000 non-null   object 
 5   Actors              1000 non-null   object 
 6   Year                1000 non-null   int64  
 7   Runtime (Minutes)   1000 non-null   int64  
 8   Rating              1000 non-null   float64
 9   Votes               1000 non-null   int64  
 10  Revenue (Millions)  872 non-null    float64
 11  Metascore           936 non-null    float64
dtypes: float64(3), int64(4), object(5)
memory usage: 93.9+ KB
```

Check for missing values

In [8]:
```
df.isnull()
```

Out[8]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...
995	False	False	False	False	False	False	False	False	False	False
996	False	False	False	False	False	False	False	False	False	False
997	False	False	False	False	False	False	False	False	False	False
998	False	False	False	False	False	False	False	False	False	False
999	False	False	False	False	False	False	False	False	False	False

1000 rows × 12 columns

In [9]:
```
df.isnull().sum()
```

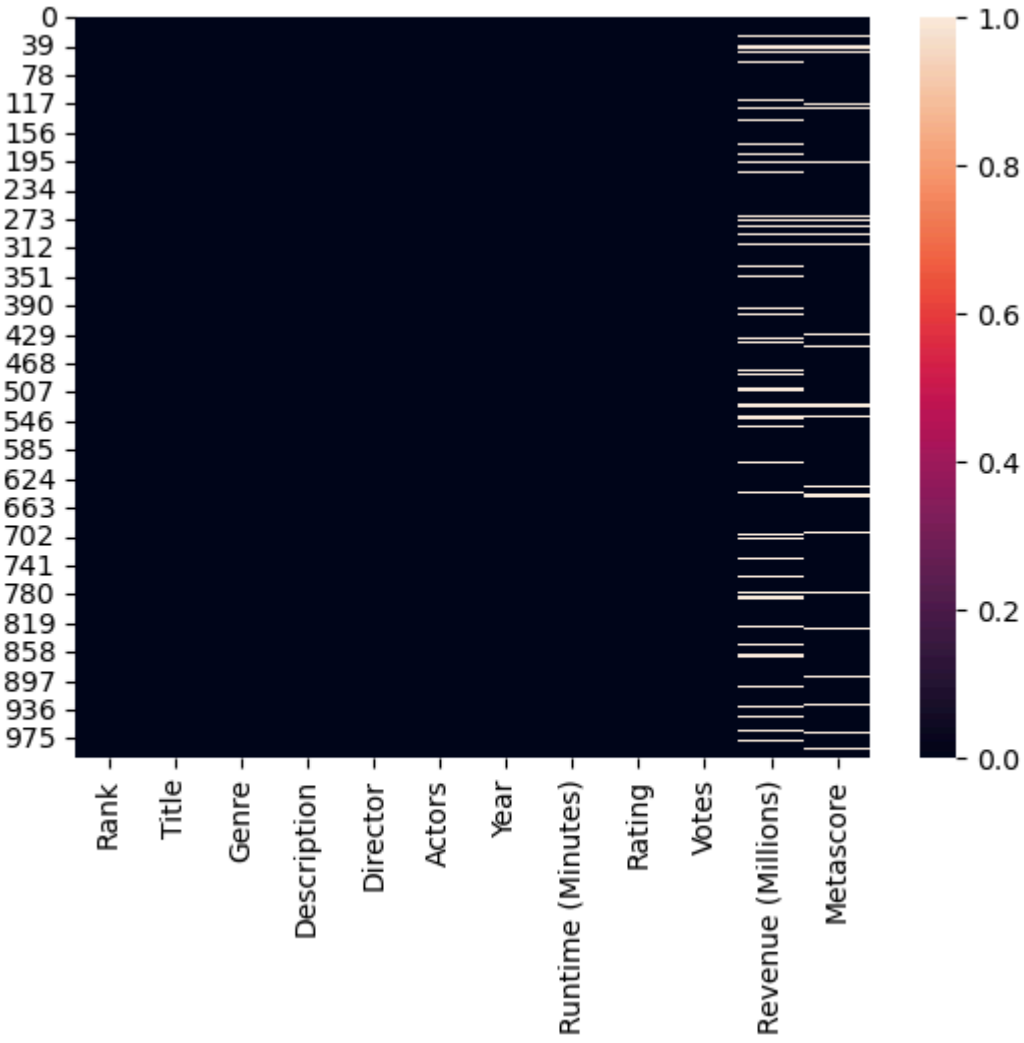
Out[9]:
```
Rank                    0
Title                   0
Genre                   0
Description             0
Director                0
Actors                  0
Year                    0
Runtime (Minutes)       0
Rating                  0
Votes                   0
Revenue (Millions)    128
Metascore              64
dtype: int64
```

In [10]:
```
df.isnull().values.any()
```

Out[10]:
```
np.True_
```

In [11]:
```
sns.heatmap(df.isnull())
```

Out[11]:
```
<Axes: >
```



In [12]:
```
# Find missing values in %

per_missing = df.isnull().sum() * 100 / len(df)
```

In [13]:
```
per_missing
```

Out[13]:
```
Rank                    0.0
Title                   0.0
Genre                   0.0
Description             0.0
Director                0.0
Actors                  0.0
Year                    0.0
Runtime (Minutes)       0.0
Rating                  0.0
Votes                   0.0
Revenue (Millions)     12.8
Metascore               6.4
dtype: float64
```

Drop all missing values

In [14]:
```
df.dropna(axis = 0)
```

Out[14]:

	Rank	Title	Genre	Description	Director	
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pra Diesel, B Cooper, Z
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi R Logan Ma Green, Micha
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James M Anya Tayl Haley Lu R
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	M McConaughey Witherspoo
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith Leto, M Robbie, Vi
...	...	...	...	...	...	
993	994	Resident Evil: Afterlife	Action,Adventure,Horror	While still out to destroy the evil Umbrella C...	Paul W.S. Anderson	Milla Jovov Larter, Wen M
994	995	Project X	Comedy	3 high school seniors throw a birthday party t...	Nima Nourizadeh	Thomas Oliver C Jonathan Dan
996	997	Hostel: Part II	Horror	Three American college students studying abroa...	Eli Roth	Lauren G Heather Mat Bijou
997	998	Step Up 2: The Streets	Drama,Music,Romance	Romantic sparks occur between two dance studen...	Jon M. Chu	Robert Hc Briana Evigan, Ver
999	1000	Nine Lives	Comedy,Family,Fantasy	A stuffy businessman	Barry Sonnenfeld	Kevin S Jennifer C

Rank	Title	Genre	Description	Director	
			finds himself trapped ins...		Robbie Am

838 rows × 12 columns

Check for Duplicate Data

In [15]:
```
df.duplicated().sum()
```

Out[15]:
```
np.int64(0)
```

In [16]:
```
df.duplicated().any()
```

Out[16]:
```
np.False_
```

Get Overall Statistics About The DataFrame

In [17]:
```
df.describe()
```

Out[17]:

	Rank	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	
count	1000.000000	1000.000000	1000.000000	1000.000000	1.000000e+03	872.000000	9
mean	500.500000	2012.783000	113.172000	6.723200	1.698083e+05	82.956376	
std	288.819436	3.205962	18.810908	0.945429	1.887626e+05	103.253540	
min	1.000000	2006.000000	66.000000	1.900000	6.100000e+01	0.000000	
25%	250.750000	2010.000000	100.000000	6.200000	3.630900e+04	13.270000	
50%	500.500000	2014.000000	111.000000	6.800000	1.107990e+05	47.985000	
75%	750.250000	2016.000000	123.000000	7.400000	2.399098e+05	113.715000	
max	1000.000000	2016.000000	191.000000	9.000000	1.791916e+06	936.630000	1

In [18]:
```
df.describe(include ='all')
```

Out[18]:

	Rank	Title	Genre	Description	Director	Actors	
count	1000.000000	1000	1000	1000	1000	1000	1000.0
unique	NaN	999	207	1000	644	996	
top	NaN	The Host	Action,Adventure,Sci-Fi	A stuffy businessman finds himself trapped ins...	Ridley Scott	Shia LaBeouf, Megan Fox, Josh Duhamel, Tyrese ...	
freq	NaN	2	50	1	8	2	
mean	500.500000	NaN	NaN	NaN	NaN	NaN	2012.7
std	288.819436	NaN	NaN	NaN	NaN	NaN	3.2
min	1.000000	NaN	NaN	NaN	NaN	NaN	2006.0
25%	250.750000	NaN	NaN	NaN	NaN	NaN	2010.0
50%	500.500000	NaN	NaN	NaN	NaN	NaN	2014.0
75%	750.250000	NaN	NaN	NaN	NaN	NaN	2016.0
max	1000.000000	NaN	NaN	NaN	NaN	NaN	2016.0

Display Title of The Movie Having Runtime Greater Than or equal to 180 Minutes

In [19]:
```
df.columns
```

Out[19]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')
```

In [20]:
```
df[df['Runtime (Minutes)']>=180]['Title']
```

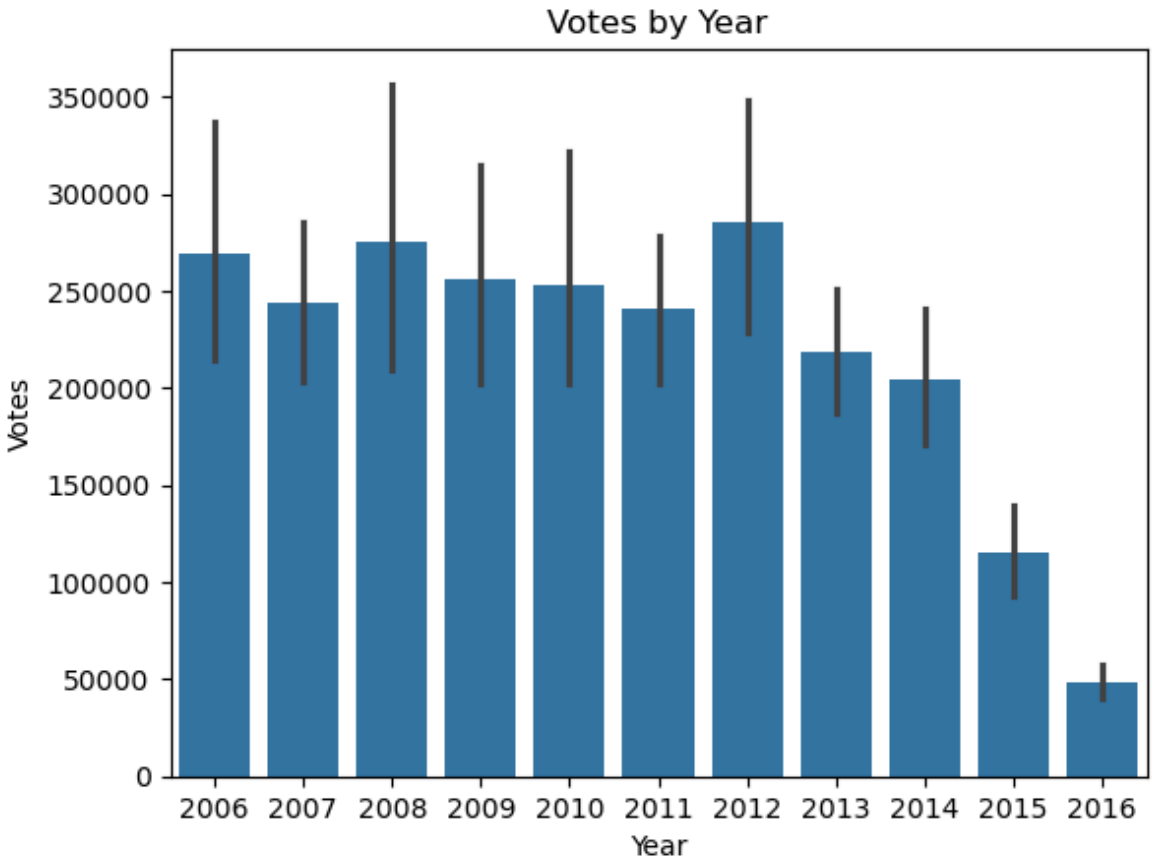
Out[20]:
```
82     The Wolf of Wall Street
88           The Hateful Eight
311             La vie d'Adèle
828                 Grindhouse
965              Inland Empire
Name: Title, dtype: object
```

In Which Year There Was The Highest Average Voting?

In [21]:
```
df.groupby('Year')['Votes'].mean().sort_values(ascending = False)
```

Out[21]:
```
Year
2012    285226.093750
2008    275505.384615
2006    269289.954545
2009    255780.647059
2010    252782.316667
2007    244331.037736
2011    240790.301587
2013    219049.648352
2014    203930.224490
2015    115726.220472
2016     48591.754209
Name: Votes, dtype: float64
```

In [22]:
```
sns.barplot(x ='Year', y ='Votes', data =df)
plt.title("Votes by Year")
plt.show()
```



In Which Year There Was The Highest Average Revenue?

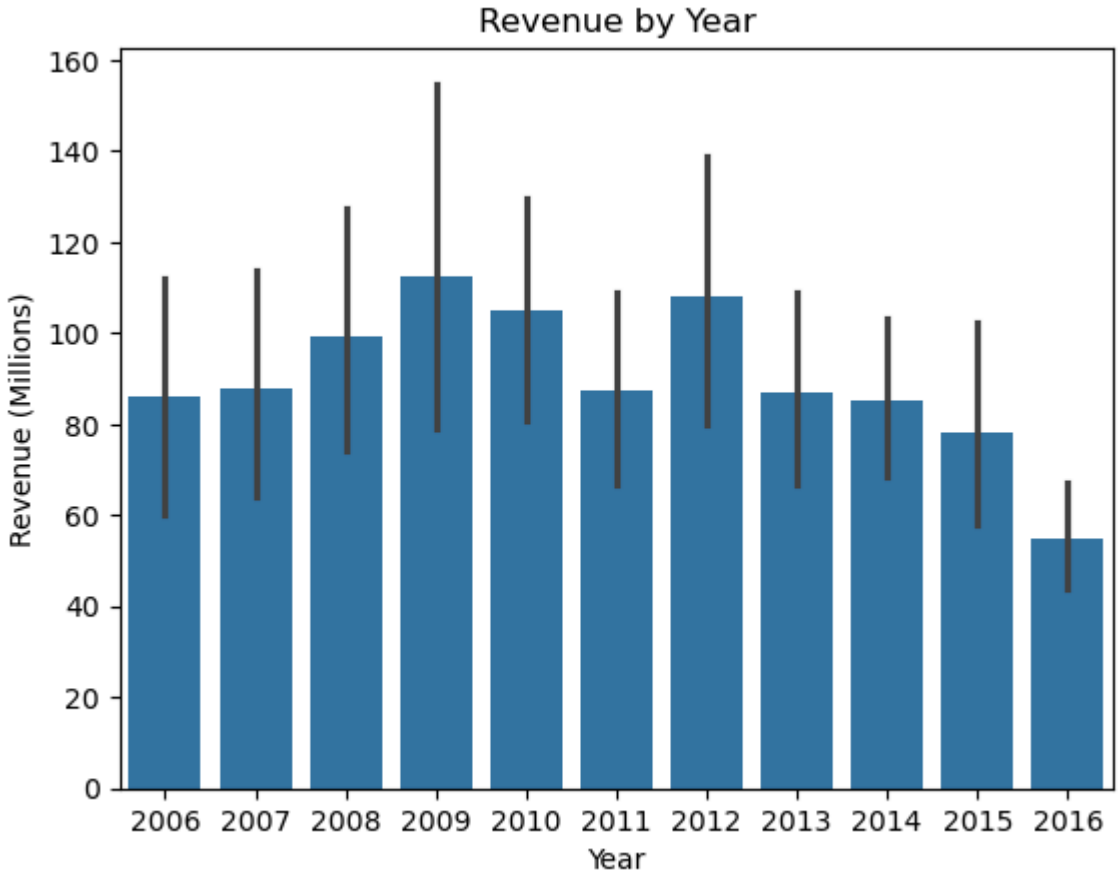
In [23]:
```
df.columns
```

Out[23]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')
```

In [24]:
```
df.groupby('Year')['Revenue (Millions)'].mean().sort_values(ascending = False)
```

Out[24]:
```
Year
2009    112.601277
2012    107.973281
2010    105.081579
2008     99.082745
2007     87.882245
2011     87.612258
2013     87.121818
2006     86.296667
2014     85.078723
2015     78.355044
2016     54.690976
Name: Revenue (Millions), dtype: float64
```

In [25]:
```
sns.barplot(x='Year' , y='Revenue (Millions)', data =df)
plt.title("Revenue by Year")
plt.show()
```



Find The Average Rating For Each Director

In [26]:
```
df.groupby("Director")["Rating"].mean().sort_values(ascending = True)
```

Out[26]:
```
Director
Jason Friedberg                    1.90
Shawn Burkett                      2.70
James Wong                         2.70
Jonathan Holbrook                  3.20
Micheal Bafaro                     3.50
                                   ...
Florian Henckel von Donnersmarck   8.50
Olivier Nakache                    8.60
Makoto Shinkai                     8.60
Christopher Nolan                  8.68
Nitesh Tiwari                      8.80
Name: Rating, Length: 644, dtype: float64
```

Display Top 10 Lengthy Movies Title and Runtime

In [27]:
```
df.columns
```

Out[27]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')
```

In [28]:
```
lengthy_movies = df.nlargest(10, 'Runtime (Minutes)') [['Title', 'Runtime (Minut
```

In [29]:
```
lengthy_movies
```

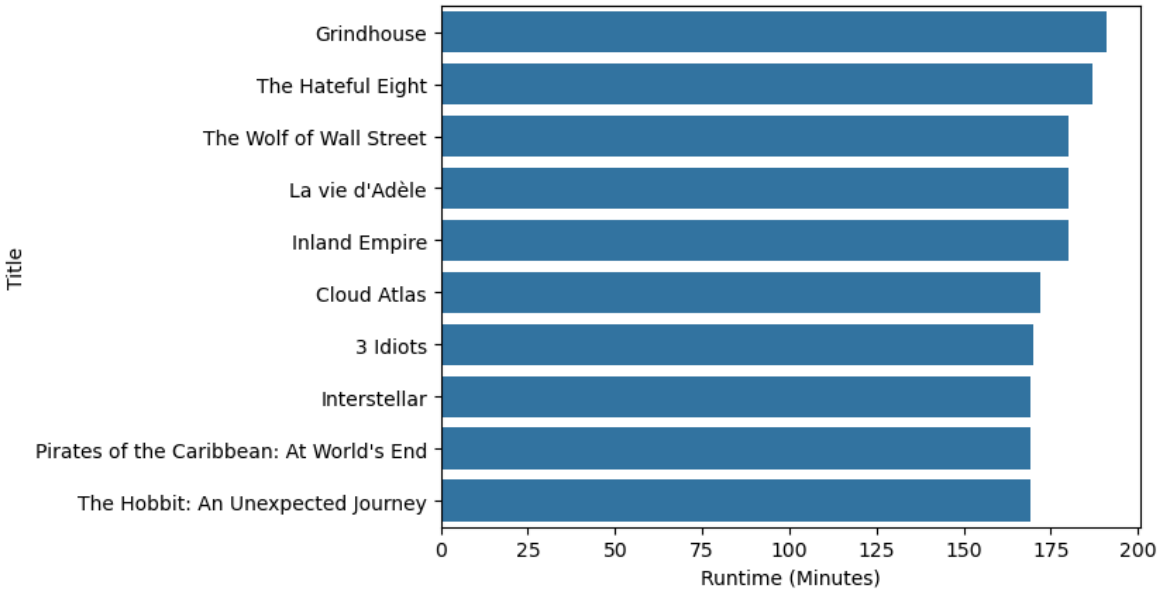
Out[29]:

Title	Runtime (Minutes)
Grindhouse	191
The Hateful Eight	187
The Wolf of Wall Street	180
La vie d'Adèle	180
Inland Empire	180
Cloud Atlas	172
3 Idiots	170
Interstellar	169
Pirates of the Caribbean: At World's End	169
The Hobbit: An Unexpected Journey	169

Display relationship between categorical data and atleast one numerical variable.

In [30]:
```
sns.barplot(x = 'Runtime (Minutes)', y = lengthy_movies.index, data = lengthy_mo
```

Out[30]: <Axes: xlabel='Runtime (Minutes)', ylabel='Title'>



Display Number of Movies Per Year

In [31]:
```
df['Year'].value_counts()   # value_counts method return object containing count
```

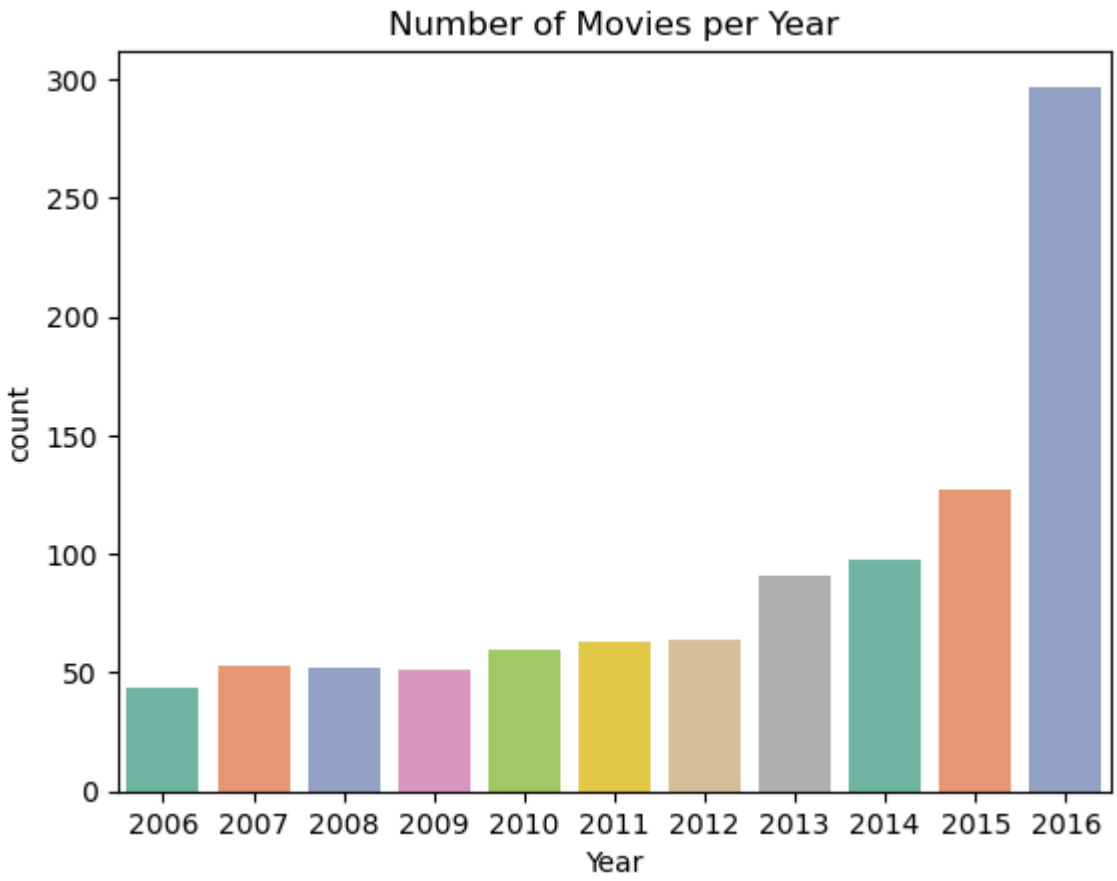
Out[31]:
```
Year
2016    297
2015    127
2014     98
2013     91
2012     64
2011     63
2010     60
2007     53
2008     52
2009     51
2006     44
Name: count, dtype: int64
```

In [34]:
```
sns.countplot(x="Year", data = df, palette='Set2')
plt.title("Number of Movies per Year")
plt.show()
```

```
C:\Users\sanad\AppData\Local\Temp\ipykernel_19864\1985436012.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.countplot(x="Year", data = df, palette='Set2')
```



Find Most Popular Movie Title (Highest Revenue)

In [35]:
```
df.columns
```

Out[35]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')
```

In [37]:
```
#  Method 1
df.groupby('Title')['Revenue (Millions)'].sum().sort_values(ascending = False)
```

Out[37]:
```
Title
Star Wars: Episode VII - The Force Awakens    936.63
Avatar                                        760.51
Jurassic World                                652.18
The Avengers                                  623.28
The Dark Knight                               533.32
                                               ...
Mindhorn                                        0.00
Martyrs                                         0.00
Wrecker                                         0.00
Mr. Nobody                                      0.00
Zipper                                          0.00
Name: Revenue (Millions), Length: 999, dtype: float64
```

In [42]:
```
# Method 2

df[df['Revenue (Millions)'].max() == df['Revenue (Millions)']]['Title']
```

Out[42]:
```
50    Star Wars: Episode VII - The Force Awakens
Name: Title, dtype: object
```

Display Top 10 Highest Rated Movie Titles And its Directors

In [47]:
```
top10 = df.nlargest(10, 'Rating')[['Title', 'Rating', 'Director']].set_index('Ti
```

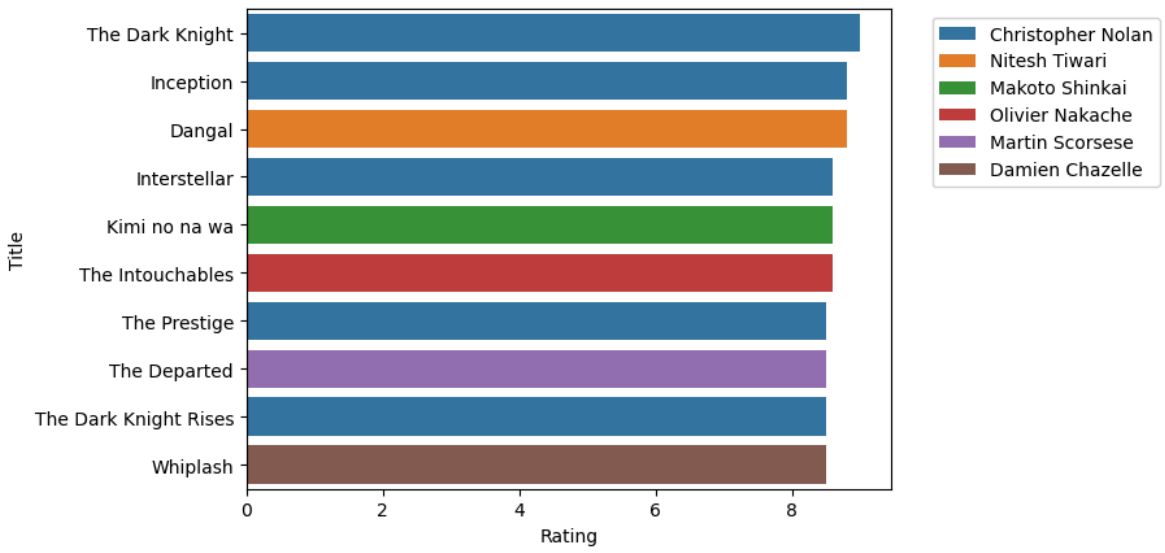
In [48]:
```
top10
```

Out[48]:

Title	Rating	Director
The Dark Knight	9.0	Christopher Nolan
Inception	8.8	Christopher Nolan
Dangal	8.8	Nitesh Tiwari
Interstellar	8.6	Christopher Nolan
Kimi no na wa	8.6	Makoto Shinkai
The Intouchables	8.6	Olivier Nakache
The Prestige	8.5	Christopher Nolan
The Departed	8.5	Martin Scorsese
The Dark Knight Rises	8.5	Christopher Nolan
Whiplash	8.5	Damien Chazelle

In [51]:
```
sns.barplot(x="Rating", y = top10.index, data = top10,
            hue="Director", dodge =False) # dodge =False - line looks better
plt.legend(bbox_to_anchor=(1.05,1),loc=2) # "bbox_to_anchor" - to position legen
```

Out[51]:
```
<matplotlib.legend.Legend at 0x2d51f429f90>
```



Display Top 10 Highest Revenue Movie Titles

In [52]:
```
df.columns
```

Out[52]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')
```

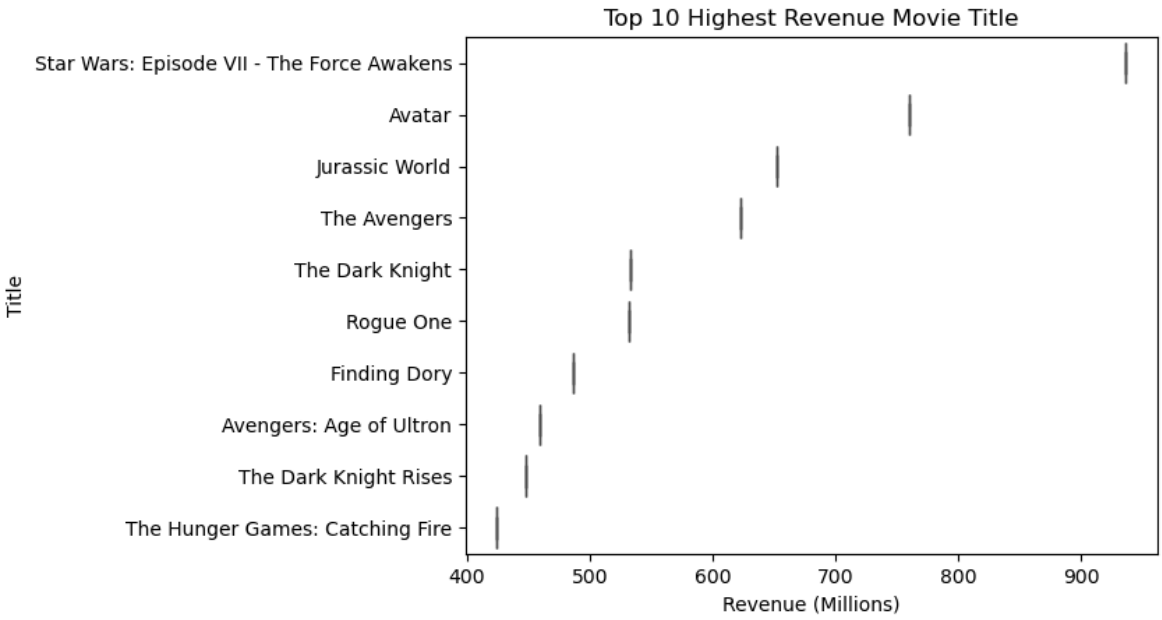
In [53]:
```
top10_revenue = df.nlargest(10, 'Revenue (Millions)')[['Title','Revenue (Million
```

In [54]:
```
top10_revenue
```

Out[54]:

Title	Revenue (Millions)
Star Wars: Episode VII - The Force Awakens	936.63
Avatar	760.51
Jurassic World	652.18
The Avengers	623.28
The Dark Knight	533.32
Rogue One	532.17
Finding Dory	486.29
Avengers: Age of Ultron	458.99
The Dark Knight Rises	448.13
The Hunger Games: Catching Fire	424.65

In [64]:
```
sns.boxplot(x='Revenue (Millions)', y=top10_revenue.index, data = top10_revenue,
plt.title("Top 10 Highest Revenue Movie Title")
plt.show()
```



Find Average Rating of Movies Year Wise

In [65]:
```
df.groupby('Year')["Rating"].mean().sort_values(ascending =False)
```

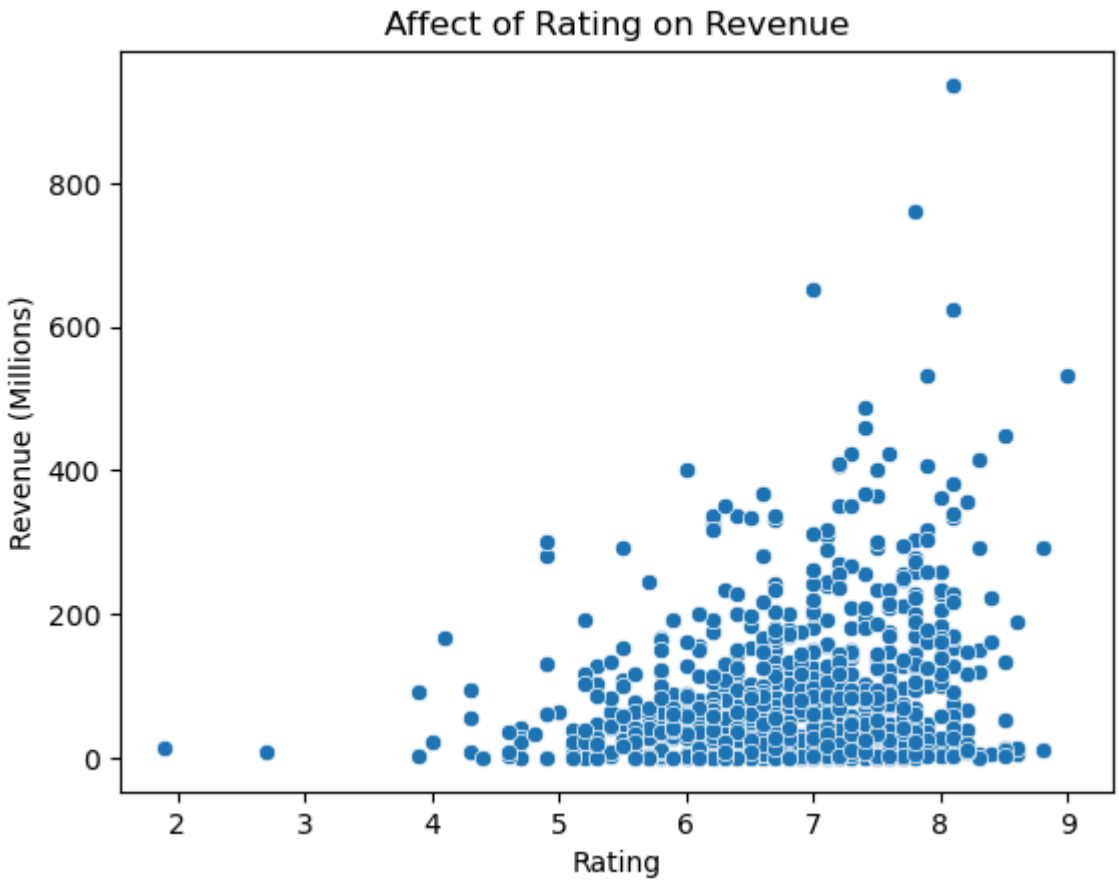
Out[65]:
```
Year
2007    7.133962
2006    7.125000
2009    6.960784
2012    6.925000
2011    6.838095
2014    6.837755
2010    6.826667
2013    6.812088
2008    6.784615
2015    6.602362
2016    6.436700
Name: Rating, dtype: float64
```

Does Rating Affect The Revenue?

In [66]:
```
df.columns
```

Out[66]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')
```

In [67]:
```
# use scatter plot to analyze the affect of rating on revenue
sns.scatterplot(x="Rating", y="Revenue (Millions)", data = df)
plt.title("Affect of Rating on Revenue")
plt.show()
```



Rating does affect revenue. Higher the rating, Revenue is also higher.

Classify Movies Based on Ratings [Excellent, Good, and Average]

In [68]:
```
df.columns
```

Out[68]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')
```

In [69]:
```
# we are creating our own function to classify Movies based on Ratings.

# def rating - function name
# (rating) - let me pass argument

def rating(rating):
    if rating>=7.0:
        return "Excellent"
    elif rating>=6.0:
        return "Good"
    else:
        return "Average"
```

In [70]:
```
df['rating_cat']=df['Rating'].apply(rating)

# you can use this created UDF using "apply" method
# lets create new column for this newly created category. "data['rating_cat']="
```

In [71]:
```
df.head()
```

Out[71]:

	Rank	Title	Genre	Description	Director	Acto
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, V Diesel, Brad Cooper, Zoe
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapa Logan Marsha Green, Michael F
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAv Anya Taylor-J Haley Lu Richa
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matth McConaughey,Ree Witherspoon, Se M
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jar Leto, Marg Robbie, Viola I

Count Number of Action Movies

In [72]:
```
df.columns
```

Out[72]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore', 'rating_cat'],
      dtype='object')
```

In [73]:
```
df['Genre'].dtypes
```

Out[73]:
```
dtype('O')
```

In [75]:
```
len(df[df['Genre'].str.contains('Action', case= False)])
```

Out[75]:
```
303
```

Find Unique values from Genre

In [76]:
```
df.Genre
```

Out[76]:
```
0       Action,Adventure,Sci-Fi
1      Adventure,Mystery,Sci-Fi
2                Horror,Thriller
3       Animation,Comedy,Family
4      Action,Adventure,Fantasy
                  ...
995         Crime,Drama,Mystery
996                      Horror
997         Drama,Music,Romance
998           Adventure,Comedy
999       Comedy,Family,Fantasy
Name: Genre, Length: 1000, dtype: object
```

Need to perform some steps to find unique values in Genre Column

- Split item with commas
- Convert this 2D list to 1D list to get unique values
- Find Unique values in empty list

In [79]:
```
list1=[]
for value in df['Genre']:
    list1.append(value.split(',')) # split the items by comms, then append the v
```

In [ ]:
```
list1
```

In [81]:
```
# Convert 2D list to 1D list
one_d=[]
for item in list1:
    for item1 in item:
        one_d.append(item1)
```

In [ ]:
```
one_d
```

In [85]:
```
# Find unique values in the empty list

uni_list=[]
for item in one_d:
    if item not in uni_list:
        uni_list.append(item)
```

In [91]:
```
uni_list
```

Out[91]:
```
['Action',
 'Adventure',
 'Sci-Fi',
 'Mystery',
 'Horror',
 'Thriller',
 'Animation',
 'Comedy',
 'Family',
 'Fantasy',
 'Drama',
 'Music',
 'Biography',
 'Romance',
 'History',
 'Crime',
 'Western',
 'War',
 'Musical',
 'Sport']
```