IMDb MOVIES ANALYSIS

Project Description

The name of the project IMDb movie analysis. Movies are the part of entertainment for the people. There are lots of movies released every year but not all the movies are hit, there are only few which are successful and rated high. There are many websites in the internet for rating the movies but one of them, IMDb is most popular among all.

Scope of Project

We have the data for the 100 top-rated movies from the past decade along with various information’s about the movie like, its actors, and the voters who has rated these movies online on IMDb. In this project, We will try to find some interesting insights into these movies and their voters, using Jupyter notebook Python.

Tech Stack Used

- Python
- Jupyter Notebook
- Microsoft Word

WHY Analysis

What do you see happening?

In IMDb movie analysis we are showing a decreasing trend in viewership, leading to lower ratings and reviews.

What is your hypothesis for the cause of the problem?

My hypothesis for the cause of the problem is that the movie is not engaging viewers and is not up to their expectations, and a decreasing trend in viewership.

What is the impact of the problem on stakeholders?

The impact of the problem on stakeholders is a decrease in revenue from the movie due to lower viewership. This can also lead to a decrease in the reputation of the movie.

What is the impact of the problem not being solved?

If the problem is not solved, it is likely that the movie will not be able to recover, resulting in a permanent decrease in viewership and ratings.

Insights

Let down the insights and the knowledge you gained while making the project. You need to write that what do you infer about the thing required to provide a detailed report for the below data record mentioning the answers of the questions that follows

- Cleaning of the data
- Find the movies with the highest profit
- Find IMDb top 250 Find top 10 directors
- Find popular genres
- Find the critic-favourite and audience-favourite actors

1. Clean and Drop the Data

- ### Read the File

To answer the above insights, we need to read the data to understand the data records for analysis for which we will be use.

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

```
In [2]: movies = pd.read_csv(r'C:\Users\hp\Desktop\IMDB_Movies.csv', encoding='ISO-8859-1')
        # movies = pd.read_csv(r"C:\Users\hp\Desktop\IMDB_Movies.csv")
        movies.head()
```

Out[2]:

	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name	actor_1_facebook_likes	gross
0	Color	James Cameron	723.0	178.0	0.0	855.0	Joel David Moore	1000.0	760505847.0
1	Color	Gore Verbinski	302.0	169.0	563.0	1000.0	Orlando Bloom	40000.0	309404152.0
2	Color	Sam Mendes	602.0	148.0	0.0	161.0	Rory Kinnear	11000.0	200074175.0
3	Color	Christopher Nolan	813.0	164.0	22000.0	23000.0	Christian Bale	27000.0	448130642.0
4	NaN	Doug Walker	NaN	NaN	131.0	NaN	Rob Walker	131.0	NaN

5 rows × 28 columns

Clean and Dropping Data

Cleaning and dropping Data is one of the most important steps to perform before moving forward with the analysis. In this step we are Dropping columns, removing null values, Sorting values, etc.

- ### Cleaning the Data

```
In [14]: # Cleaning The Data

         movies.isnull().sum(axis=1).sort_values(ascending= False)

Out[14]: 279     16
         4       14
         2241    12
         4945    12
         2342    11
                 ..
         2708     0
         2707     0
         2706     0
         2705     0
         0        0
         Length: 5043, dtype: int64
```

- ### Removing Null Values

```
In [15]: # Removing Null Values
         movies.isnull().sum().sort_values(ascending= False)/len(movies)*100

Out[15]: profit                       22.843546
         gross                        17.529248
         budget                        9.756098
         aspect_ratio                  6.523895
         content_rating                6.008328
         plot_keywords                 3.033908
         title_year                    2.141582
         director_name                 2.062265
         director_facebook_likes       2.062265
         num_critic_for_reviews        0.991473
         actor_3_name                  0.456078
         actor_3_facebook_likes        0.456078
         num_user_for_reviews          0.396589
         color                         0.376760
         duration                      0.297442
         actor_2_name                  0.257783
         facenumber_in_poster          0.257783
         actor_2_facebook_likes        0.257783
         language                      0.237954
         actor_1_name                  0.138806
         actor_1_facebook_likes        0.138806
         country                       0.099147
         cast_total_facebook_likes     0.000000
         num_voted_users               0.000000
         movie_title                   0.000000
         genres                        0.000000
         genre_1                       0.000000
         movie_imdb_link               0.000000
         imdb score                    0.000000
```

- ### Dropping the Data

In [16]:
```python
# Dropping The Data

movies=movies.drop(['color','director_facebook_likes','actor_1_facebook_likes','actor_2_facebook_likes','actor_3_name','actor_2_r
movies
```

Out[16]:

	director_name	num_critic_for_reviews	actor_3_facebook_likes	gross	genres	actor_1_name	movie_title	num_voted_users	cast_
0	James Cameron	723.0	855.0	760.505847	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	AvatarÂ	886204	
1	Gore Verbinski	302.0	1000.0	309.404152	Action|Adventure|Fantasy	Johnny Depp	Pirates of the Caribbean: At World's EndÂ	471220	
2	Sam Mendes	602.0	161.0	200.074175	Action|Adventure|Thriller	Christoph Waltz	SpectreÂ	275868	
3	Christopher Nolan	813.0	23000.0	448.130642	Action|Thriller	Tom Hardy	The Dark Knight RisesÂ	1144337	
4	Doug Walker	NaN	NaN	NaN	Documentary	Doug Walker	Star Wars: Episode VII - The Force AwakensÂ ...	8	
...	...	...	...	...	...	...	...	...	
5038	Scott Smith	1.0	318.0	NaN	Comedy|Drama	Eric Mabius	Signed Sealed DeliveredÂ	629	
5039	NaN	43.0	319.0	NaN	Crime|Drama|Mystery|Thriller	Natalie Zea	The [illegible]	73839	

- ## Sorting the Data

In [17]:
```python
# sorting the data

round(movies.isnull().sum().sort_values(ascending=False)/len(movies)*100,2)
```

Out[17]:
```
profit                     22.84
gross                      17.53
budget                      9.76
aspect_ratio                6.52
content_rating              6.01
plot_keywords               3.03
title_year                  2.14
director_name               2.06
num_critic_for_reviews      0.99
actor_3_facebook_likes      0.46
num_user_for_reviews        0.40
language                    0.24
actor_1_name                0.14
country                     0.10
genre_1                     0.00
movie_imdb_link             0.00
cast_total_facebook_likes   0.00
num_voted_users             0.00
movie_title                 0.00
genres                      0.00
imdb_score                  0.00
movie_facebook_likes        0.00
genre_2                     0.00
dtype: float64
```

2. Movies with Highest Profit

Creating a new column called profit which contains the difference of the two columns: gross and budget.

In [19]:
```python
# Movies with Highest profit

movies['budget']=movies['budget']/1000000
movies['gross']=movies['gross']/1000000
movies['profit']=movies['gross']-movies['budget']
movies
```

Out[19]:

	director_name	num_critic_for_reviews	actor_3_facebook_likes	gross	genres	actor_1_name	movie_title	num_voted_users	ca
0	James Cameron	723.0	855.0	7.605058e-10	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	AvatarÂ	886204	
1	Gore Verbinski	302.0	1000.0	3.094042e-10	Action|Adventure|Fantasy	Johnny Depp	Pirates of the Caribbean: At World's EndÂ	471220	
2	Sam Mendes	602.0	161.0	2.000742e-10	Action|Adventure|Thriller	Christoph Waltz	SpectreÂ	275868	
3	Christopher Nolan	813.0	23000.0	4.481306e-10	Action|Thriller	Tom Hardy	The Dark Knight RisesÂ	1144337	
4	Doug Walker	NaN	NaN	NaN	Documentary	Doug Walker	Star Wars: Episode VII - The Force AwakensÂ	8	

Sorting the column using the profit column as reference, Top 5 Movies with Highest Profit

In [20]:
```python
# Top 5 Highest Profitable Movies

top_5 = movies.sort_values(by='profit', ascending=False).head(5)
top_5
```

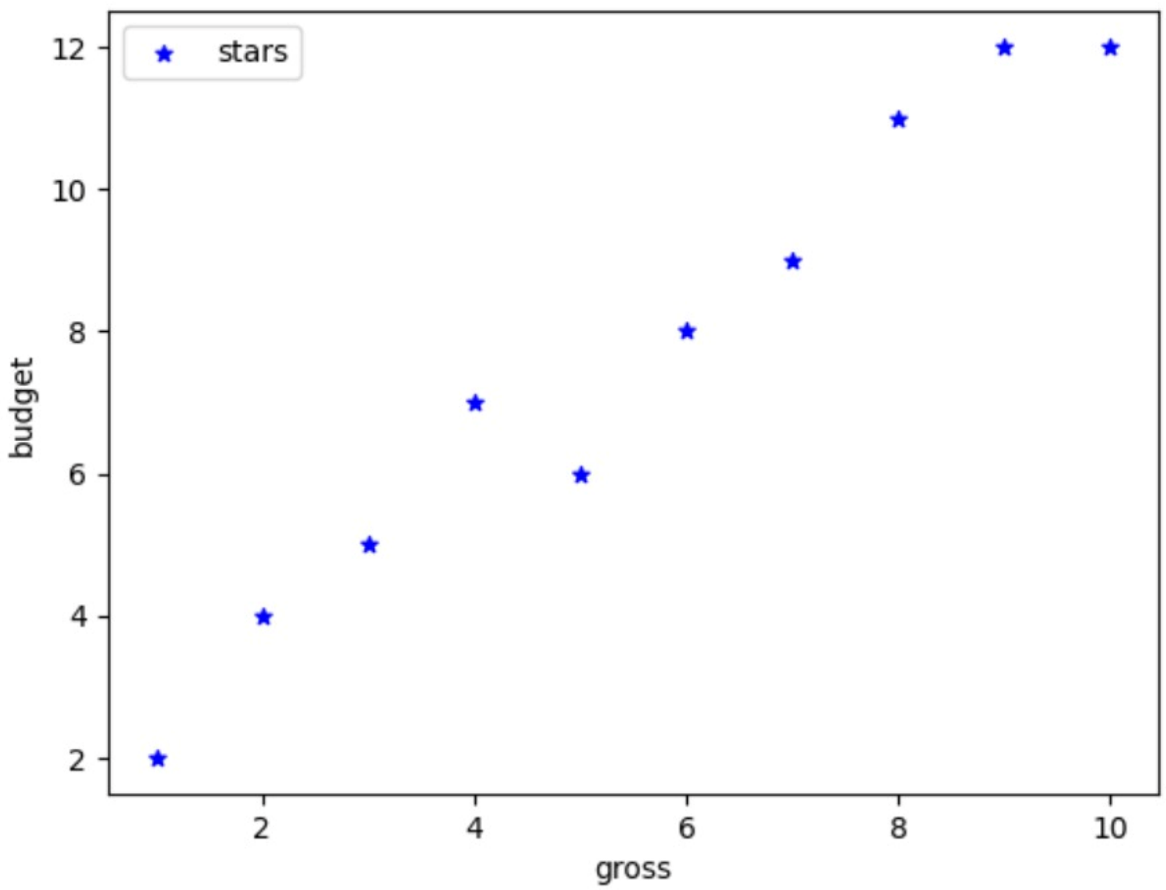
Out[20]:

	director_name	num_critic_for_reviews	actor_3_facebook_likes	gross	genres	actor_1_name	movie_title	num_voted_users	cast_
0	James Cameron	723.0	855.0	7.605058e-10	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	AvatarÂ	886204	
29	Colin Trevorrow	644.0	1000.0	6.521773e-10	Action|Adventure|Sci-Fi|Thriller	Bryce Dallas Howard	Jurassic WorldÂ	418214	
26	James Cameron	315.0	794.0	6.586723e-10	Drama|Romance	Leonardo DiCaprio	TitanicÂ	793059	
3024	George Lucas	282.0	504.0	4.609357e-10	Action|Adventure|Fantasy|Sci-Fi	Harrison Ford	Star Wars: Episode IV - A New HopeÂ	911097	
3080	Steven Spielberg	215.0	548.0	4.349495e-10	Family|Sci-Fi	Henry Thomas	E.T. the Extra-TerrestrialÂ	281842	

5 rows × 23 columns

Plotting profit (y-axis) vs budget (x- axis) and here we observe the outliers using the appropriate chart type.

Budget Vs Gross Graph



3. IMDb Top 250

Creating a new column IMDb_Top_250 and storing the top 250 movies with the highest IMDb Rating.

```
In [21]: # Top 250 imdb movies

         movies[movies['num_voted_users']>25000].sort_values(by='imdb_score', ascending = False)
```

Out[21]:

	director_name	num_critic_for_reviews	actor_3_facebook_likes	gross	genres	actor_1_name	movie_title	num_voted_users	cast_t
1937	Frank Darabont	199.0	461.0	2.834147e-11	Crime|Drama	Morgan Freeman	The Shawshank RedemptionÂ	1689764	
3466	Francis Ford Coppola	208.0	3000.0	1.348220e-10	Crime|Drama	Al Pacino	The GodfatherÂ	1155770	
3481	NaN	54.0	1000.0	NaN	Crime|Drama|Thriller	Kirsten Dunst	FargoÂ	170055	
66	Christopher Nolan	645.0	11000.0	5.333161e-10	Action|Crime|Drama|Thriller	Christian Bale	The Dark KnightÂ	1676169	
2837	Francis Ford Coppola	149.0	3000.0	5.730000e-11	Crime|Drama	Robert De Niro	The Godfather: Part IIÂ	790926	
...	...	...	...	...	...	...	...	...	
2192	Jason Friedberg	112.0	729.0	3.973765e-11	Adventure|Comedy	David Carradine	Epic MovieÂ	89687	
319	Lawrence Guterman	78.0	227.0	1.701065e-11	Comedy|Family|Fantasy	Jamie Kennedy	Son of the MaskÂ	40751	
2295	Bob Clark	32.0	177.0	9.109322e-12	Comedy|Family|Sci-Fi	Scott Baio	Superbabies: Baby Geniuses 2Â	25371	
2268	Jason Friedberg	111.0	329.0	1.417465e-11	Comedy	Carmen Electra	Disaster MovieÂ	74945	

The num_voted_users is Greater than 25,000. Also adding a Rank column containing the values 1 to 250 indicating the ranks of the corresponding films.

```
In [22]: # num_voted_users is greater than 25000

         top_250= movies[movies['num_voted_users']>25000].sort_values(by='imdb_score', ascending = False).head(250)
         top_250
```

Out[22]:

	director_name	num_critic_for_reviews	actor_3_facebook_likes	gross	genres	actor_1_name	movie_title	num_voted_users	cast
1937	Frank Darabont	199.0	461.0	2.834147e-11	Crime|Drama	Morgan Freeman	The Shawshank RedemptionÂ	1689764	
3466	Francis Ford Coppola	208.0	3000.0	1.348220e-10	Crime|Drama	Al Pacino	The GodfatherÂ	1155770	
3481	NaN	54.0	1000.0	NaN	Crime|Drama|Thriller	Kirsten Dunst	FargoÂ	170055	
66	Christopher Nolan	645.0	11000.0	5.333161e-10	Action|Crime|Drama|Thriller	Christian Bale	The Dark KnightÂ	1676169	
2837	Francis Ford Coppola	149.0	3000.0	5.730000e-11	Crime|Drama	Robert De Niro	The Godfather: Part IIÂ	790926	
...	...	...	...	...	...	...	...	...	
4266	Richard Linklater	211.0	3.0	5.792822e-12	Drama|Romance	Vernon Dobtcheff	Before SunsetÂ	168398	
602	Tim Burton	235.0	883.0	6.625700e-11	Adventure|Drama|Fantasy	Steve Buscemi	Big FishÂ	350698	
1603	Clint Eastwood	229.0	262.0	9.013519e-11	Crime|Drama|Mystery|Thriller	John Doman	Mystic RiverÂ	338415	
4261	Robert Rossen	100.0	366.0	NaN	Drama|Sport	George C. Scott	The HustlerÂ	62860	
1601	Neill Blomkamp	472.0	107.0	1.156462e-10	Action|Sci-Fi|Thriller	Sharlto Copley	District 9Â	531737	

250 rows × 23 columns

Extracting all the movies in the IMDb_Top_250 column which are not in the English language and store them in a new column named Top_Foreign_Lang_Film.

```
In [23]: # Top 250 movies which are not in english Language

         top_250[top_250['language']!='English']
```

Out[23]:

	director_name	num_critic_for_reviews	actor_3_facebook_likes	gross	genres	actor_1_name	movie_title	n
4498	Sergio Leone	181.0	24.0	6.100000e-12	Western	Clint Eastwood	The Good, the Bad and the UglyÂ	
4747	Akira Kurosawa	153.0	4.0	2.690610e-13	Action|Adventure|Drama	Takashi Shimura	Seven SamuraiÂ	
4029	Fernando Meirelles	214.0	40.0	7.563397e-12	Crime|Drama	Alice Braga	City of GodÂ	
2373	Hayao Miyazaki	246.0	7.0	1.004989e-11	Adventure|Animation|Family|Fantasy	Bunta Sugawara	Spirited AwayÂ	
3870	Raja Menon	39.0	12.0	NaN	Action|Drama|History|Thriller|War	Nimrat Kaur	AirliftÂ	
4259	Florian Henckel von Donnersmarck	215.0	155.0	1.128466e-11	Drama|Thriller	Sebastian Koch	The Lives of OthersÂ	
4921	Majid Majidi	46.0	27.0	9.254020e-13	Drama|Family	Bahare Seddiqi	Children of HeavenÂ	
4105	Chan-wook Park	305.0	38.0	2.181290e-12	Drama|Mystery|Thriller	Min-sik Choi	OldboyÂ	
4659	Asghar Farhadi	354.0	620.0	7.098492e-12	Drama|Mystery	Shahab Hosseini	A SeparationÂ	
3685	Rakeysh Omprakash Mehra	33.0	199.0	2.197331e-12	Comedy|Drama|History|Romance	Anupam Kher	Rang De BasantiÂ	
2970	Wolfgang Petersen	96.0	18.0	1.143313e-11	Adventure|Drama|Thriller|War	JÃ¼rgen Prochnow	Das BootÂ	

4. Top 10 Directors

Grouping the column using the director_name column.

Best Directors

```
In [24]: # Best Director

         movies.groupby('director_name').imdb_score.mean().sort_values(ascending=False)

Out[24]: director_name
         John Blanchard          9.5
         Sadyk Sher-Niyaz        8.7
         Mitchell Altieri        8.7
         Cary Bell               8.7
         Mike Mayhall            8.6
                                ...
         Georgia Hilton          2.2
         Vondie Curtis-Hall      2.1
         FrÃ©dÃ©ric Auburtin     2.0
         A. Raven Cruz           1.9
         Lawrence Kasanoff       1.7
         Name: imdb_score, Length: 2398, dtype: float64
```

Top 10 Directors

To find out the top 10 directors for whom the mean of imdb_score is the highest and store them in a new column named as Top_10_directors. If in case of same in IMDb score occurs between two directors, sorting them alphabetically

In [9]:

```python
# top 10 Director

top_10_directors = movies.groupby('director_name').imdb_score.mean().sort_values(ascending=False).head(10)
top_10_directors
```

Out[9]:

```
director_name
John Blanchard        9.5
Sadyk Sher-Niyaz      8.7
Mitchell Altieri      8.7
Cary Bell             8.7
Mike Mayhall          8.6
Charles Chaplin       8.6
Ron Fricke            8.5
Majid Majidi          8.5
Raja Menon            8.5
Damien Chazelle       8.5
Name: imdb_score, dtype: float64
```

5. Popular Genres

Grouping the column using the Genre_1 and Genre_2 column.

In [40]:

```python
# popular genres

genre=movies.genres.str.split('|',expand=True).iloc[:,0:2]
genre.columns=['genre_1','genre_2']
genre.genre_2.fillna(genre.genre_1,inplace=True)
genre
```

Out[40]:

	genre_1	genre_2
0	Action	Adventure
1	Action	Adventure
2	Action	Adventure
3	Action	Thriller
4	Documentary	Documentary
...	...	...
5038	Comedy	Drama
5039	Crime	Drama
5040	Drama	Horror
5041	Comedy	Drama
5042	Documentary	Documentary

5043 rows × 2 columns

To find out the Top 5 Popular Genres and store them in a new column.

In [41]:

```python
# Top 5 Popular Genres
movies=pd.concat([movies, genre],axis=1)
movies
```

Out[41]:

	director_name	num_critic_for_reviews	actor_3_facebook_likes	gross	genres	actor_1_name	movie_title	num_voted_users	cast_
0	James Cameron	723.0	855.0	7.605058e-10	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	AvatarÂ	886204	
1	Gore Verbinski	302.0	1000.0	3.094042e-10	Action|Adventure|Fantasy	Johnny Depp	Pirates of the Caribbean: At World's EndÂ	471220	
2	Sam Mendes	602.0	161.0	2.000742e-10	Action|Adventure|Thriller	Christoph Waltz	SpectreÂ	275868	
3	Christopher Nolan	813.0	23000.0	4.481306e-10	Action|Thriller	Tom Hardy	The Dark Knight RisesÂ	1144337	
4	Doug Walker	NaN	NaN	NaN	Documentary	Doug Walker	Star Wars: Episode VII - The Force AwakensÂ ...	8	
...	...	...	...	...	...	...	...	...	
5038	Scott Smith	1.0	318.0	NaN	Comedy|Drama	Eric Mabius	Signed Sealed DeliveredÂ	629	
5039	NaN	43.0	319.0	NaN	Crime|Drama|Mystery|Thriller	Natalie Zea	The FollowingÂ	73839	
5040	Benjamin Roberds	13.0	0.0	NaN	Drama|Horror|Thriller	Eva Boehnke	A Plague So PleasantÂ	38	
5041	Daniel Hsia	14.0	489.0	1.044300e-	Comedy|Drama|Romance	Alan Ruck	Shanghai	1255	

Sorting Top 5 Popular Genres Alphabetically

```
In [18]: # Top 5 Popular genres alphabetically

         movies.groupby(['genre_1','genre_2']).gross.mean().sort_values(ascending=False).head(5)
         # movies

Out[18]: genre_1    genre_2
         Family     Sci-Fi       434.949459
         Adventure  Sci-Fi       228.627758
                    Animation    117.050005
                    Family       113.299411
         Action     Adventure    109.540018
         Name: gross, dtype: float64
```

6. Critic - Favourite and Audience - Favourite Actors

We Create three new columns namely, Meryl_Streep, Leo_Caprio, and Brad_Pitt which contain the movies in which the actors: 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' are the lead actors. Using only the actor_1_name column for extraction. Also, making sure that we use the names 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' for the said extraction.

Appending the rows of all these columns and store them in a new column named Combined.

Group the combined column using the actor_1_name column

```
In [48]: # combined column using the actor_1_name

         Meryl_streep = movies[movies['actor_1_name']=='Meryl Streep']
         Leo_Caprio = movies[movies['actor_1_name']=='Leonardo DiCaprio']
         Brad_Pitt = movies[movies['actor_1_name']=='Brad Pitt']

In [49]: combined= Meryl_streep.append([Leo_Caprio, Brad_Pitt])
         combined

Out[49]:
```

	director_name	num_critic_for_reviews	actor_3_facebook_likes	gross	genres	actor_1_name	movie_title
410	Nancy Meyers	187.0	963.0	1.127035e-10	Comedy|Drama|Romance	Meryl Streep	It's ComplicatedÂ
1106	Curtis Hanson	42.0	132.0	4.681575e-11	Action|Adventure|Crime|Thriller	Meryl Streep	The River WildÂ
1204	Nora Ephron	252.0	923.0	9.412543e-11	Biography|Drama|Romance	Meryl Streep	Julie & JuliaÂ
1408	David Frankel	208.0	505.0	1.247330e-10	Comedy|Drama|Romance	Meryl Streep	The Devil Wears PradaÂ
1483	Robert Redford	227.0	10000.0	1.499807e-11	Drama|Thriller|War	Meryl Streep	Lions for LambsÂ
1575	Sydney Pollack	66.0	184.0	8.710000e-11	Biography|Drama|Romance	Meryl Streep	Out of AfricaÂ
1618	David Frankel	234.0	329.0	6.353601e-[illegible]	Comedy|Drama|Romance	Meryl Streep	Hope [illegible]

Finding the mean of the num_critic_for_reviews and num_users_for_review and identify the actors which have the highest mean.

Number of Critic Reviews

```
In [45]: # Number of critics for review

         combined.groupby('actor_1_name').num_critic_for_reviews.mean()

Out[45]: actor_1_name
         Brad Pitt       231.944444
         Meryl Streep    163.153846
         Name: num_critic_for_reviews, dtype: float64
```

Number of User Reviews

```
In [50]: # Number of User Reviews

         combined.num_user_for_reviews= combined.num_user_for_reviews.astype('int')
         combined.num_user_for_reviews

Out[50]: 410      214
         1106      69
         1204     277
         1408     631
         1483     298
         1575     200
         1618     178
         1674     112
         1752      32
         1925     660
         2781     350
         3135     280
         3641      44
         26      2528
         50       753
         97      2803
         179     1188
         257      799
         296     1193
         307      657
         308     1138
         326     1166
         361     2054
         452      964
         641      263
         911      667
         990      548
         1114     414
         1422     244
         1453     279
```

```
In [51]: combined.groupby('actor_1_name').num_user_for_reviews.mean()

Out[51]: actor_1_name
         Brad Pitt            702.444444
         Leonardo DiCaprio    914.476190
         Meryl Streep         257.307692
         Name: num_user_for_reviews, dtype: float64
```

Actors Having Highest Mean Value

```
In [52]: # Actors Having Highest mean Value

         combined.groupby('actor_1_name')['num_user_for_reviews', 'num_critic_for_reviews'].mean()

         <ipython-input-52-ad0b49140fe2>:3: FutureWarning: Indexing with multiple keys (implicitly converted to a tuple of keys) will be
         deprecated, use a list instead.
           combined.groupby('actor_1_name')['num_user_for_reviews', 'num_critic_for_reviews'].mean()

Out[52]:
```

actor_1_name	num_user_for_reviews	num_critic_for_reviews
Brad Pitt	702.444444	231.944444
Leonardo DiCaprio	914.476190	330.190476
Meryl Streep	257.307692	163.153846

Observe the change in number of voted users over decades using a bar chart.

Create a column called decade which represents the decade to which every movie belongs to. For example, the title_year 1923, 1925 should be stored as 1920s. Sort the column based on the column decade, group it by decade and find the sum of users voted in each decade. Store it in a new data frame called df_by_decade

```
In [58]: # Create Decade Column

         df['decade'] = (df['title_year']//10)*10


         # group by decade and find sum of users voted

         df_by_decade = df.groupby('decade', as_index=False)['num_voted_users'].sum()

         # Plotting the bar chart

         plt.bar(df_by_decade['decade'], df_by_decade['num_voted_users'])
         plt.xlabel('Decade')
         plt.ylabel('No. of Voted users')
         plt.title('No. of Voted Users by Decade')
         plt.show()
```