IMDB Movie Analysis

IMDB (Internet Movie Database) is a website that provides a comprehensive collection of information about films, TV shows, and video games. It is a popular source for movie analysis and reviews. IMDB users can rate and review movies, and these ratings are used to calculate a movie's overall score, which is displayed on the website.

IMDB Movie Analysis involves examining various aspects of a movie, including its storyline, characters, direction, cinematography, soundtrack, and overall production value. This analysis can help viewers understand the movie on a deeper level and appreciate its artistic and technical merits.

IMDB Movie Analysis typically involves analyzing a movie's plot and characters. The plot analysis examines the movie's overall story arc, its pacing, and its structure. The character analysis looks at the motivations, relationships, and development of the movie's main characters.

In addition to plot and character analysis, IMDB Movie Analysis may also examine the technical aspects of the movie, including its direction, cinematography, and editing. The analysis may also examine the movie's soundtrack, including the use of music and sound effects to enhance the emotional impact of the story.

Overall, IMDB Movie Analysis provides a way to explore the elements that make a movie great, and to appreciate the skill and artistry that goes into creating a memorable film.

My question is : What are the top movies of our generation and who are the directors and actors that worked on it and what are the reviews that they got?

The answers can be found in this report.

Task 1 : Clean the dataset

Data cleaning, also known as data cleansing or data scrubbing, is the process of identifying and correcting errors, inconsistencies, and inaccuracies in data. It is an important step in preparing data for analysis or use in a database, as it ensures that the data is accurate, complete, and consistent.

Without proper data cleaning, the results of data analysis can be skewed or even completely incorrect, leading to flawed conclusions and decisions.

Null Values present Before

```
color                         19
director_name                104
num_critic_for_reviews        50
duration                      15
director_facebook_likes      104
actor_3_facebook_likes        23
actor_2_name                  13
actor_1_facebook_likes         7
gross                        884
genres                         0
actor_1_name                   7
movie_title                    0
num_voted_users                0
cast_total_facebook_likes      0
actor_3_name                  23
facenumber_in_poster          13
plot_keywords                153
movie_imdb_link                0
num_user_for_reviews          20
language                      12
country                        5
content_rating               303
budget                       492
title_year                   108
actor_2_facebook_likes        13
imdb_score                     0
aspect_ratio                 329
movie_facebook_likes           0
dtype: int64
```

```python
data.drop(["actor_1_facebook_likes","duration","actor_3_facebook_likes","language",
           "gross","facenumber_in_poster","director_facebook_likes",'cast_total_facebook_likes','movie_facebook_likes',
           'plot_keywords',"movie_imdb_link",'actor_2_facebook_likes'],axis=1,inplace=True)
```

```python
#removing NaN value
data['color']=data['color'].fillna("color")
data.replace({"actor_2_name":np.NaN,
              "actor_3_name":np.NaN,
              "actor_1_name":np.NaN,
              "country":np.NaN,
             "content_rating":np.NaN},value="None",inplace=True)
```

```python
data.replace({'director_name':np.NaN},value="None",inplace=True)
data['num_critic_for_reviews']=data['num_critic_for_reviews'].fillna(value=data['num_critic_for_reviews'].mean())
data["aspect_ratio"]=data['aspect_ratio'].fillna(method='ffill')
data.drop(data.index[4],inplace=True)
```

```python
data['budget']=data['budget'].fillna(data['budget'].mean())
```

```python
data['num_user_for_reviews']=data['num_user_for_reviews'].astype('float64')
data['num_user_for_reviews']=data['num_user_for_reviews'].fillna(value=data['num_user_for_reviews'].mean())
```

```python
data['title_year']=data['title_year'].astype('float64')
data['title_year']=data['title_year'].fillna(value=data['num_user_for_reviews'].mean())
```

After

```
color                     0
director_name             0
num_critic_for_reviews    0
actor_2_name              0
genres                    0
actor_1_name              0
movie_title               0
num_voted_users           0
actor_3_name              0
num_user_for_reviews      0
country                   0
content_rating            0
budget                    0
title_year                0
imdb_score                0
aspect_ratio              0
dtype: int64
```

```
data.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 5042 entries, 0 to 5042
Data columns (total 16 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   color                   5042 non-null   object
 1   director_name           5042 non-null   object
 2   num_critic_for_reviews  5042 non-null   float64
 3   actor_2_name            5042 non-null   object
 4   genres                  5042 non-null   object
 5   actor_1_name            5042 non-null   object
 6   movie_title             5042 non-null   object
 7   num_voted_users         5042 non-null   int64
 8   actor_3_name            5042 non-null   object
 9   num_user_for_reviews    5042 non-null   float64
 10  country                 5042 non-null   object
 11  content_rating          5042 non-null   object
 12  budget                  5042 non-null   float64
 13  title_year              5042 non-null   float64
 14  imdb_score              5042 non-null   float64
 15  aspect_ratio            5042 non-null   float64
dtypes: float64(6), int64(1), object(9)
memory usage: 669.6+ KB
```

Task 2: Movies with Highest Profit

```
profit_data['budget']=profit_data['budget'].fillna(profit_data['budget'].mean())
profit_data['gross']=profit_data['gross'].fillna(profit_data['gross'].mean())
profit_data['Profit']=profit_data['gross']-profit_data['budget']
profit_data.drop(["actor_1_facebook_likes","duration","actor_3_facebook_likes","language",
                 "genres","actor_1_name","facenumber_in_poster","director_facebook_likes","cast_total_facebook_likes",
                 'plot_keywords','title_year','movie_imdb_link','actor_2_facebook_likes','num_user_for_reviews'],axis=1,inplace=True)
profit_data.sort_values("Profit",ascending=False)
```

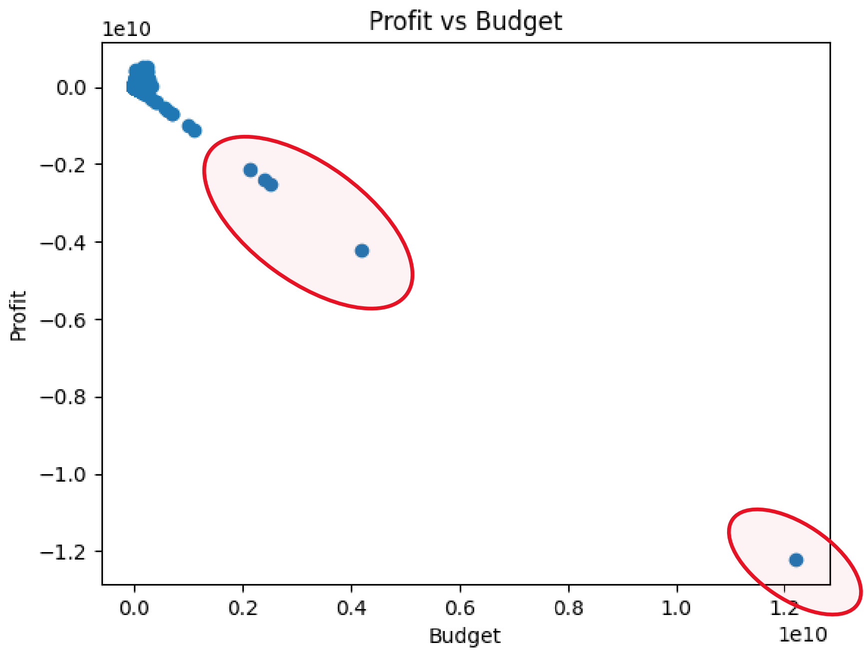
These are the movies sorted according to Profit:

	color	director_name	num_critic_for_reviews	actor_2_name	gross	movie_title
0	Color	James Cameron	723.0	Joel David Moore	760505847.0	Avatar
29	Color	Colin Trevorrow	644.0	Judy Greer	652177271.0	Jurassic World
26	Color	James Cameron	315.0	Kate Winslet	658672302.0	Titanic
3024	Color	George Lucas	282.0	Peter Cushing	460935665.0	Star Wars: Episode IV - A New Hope
3080	Color	Steven Spielberg	215.0	Dee Wallace	434949459.0	E.T. the Extra-Terrestrial
...	...	...	...	...	...	...
2334	Color	Katsuhiro Ôtomo	105.0	Robin Atkin Downes	410388.0	Steamboy
2323	Color	Hayao Miyazaki	174.0	Jada Pinkett Smith	2298191.0	Princess Mononoke
3005	Color	Lajos Koltai	73.0	Péter Fancsikai	195888.0	Fateless
3859	Color	Chan-wook Park	202.0	Yeong-ae Lee	211667.0	Lady Vengeance
2988	Color	Joon-ho Bong	363.0	Kang-ho Song	2201412.0	The Host

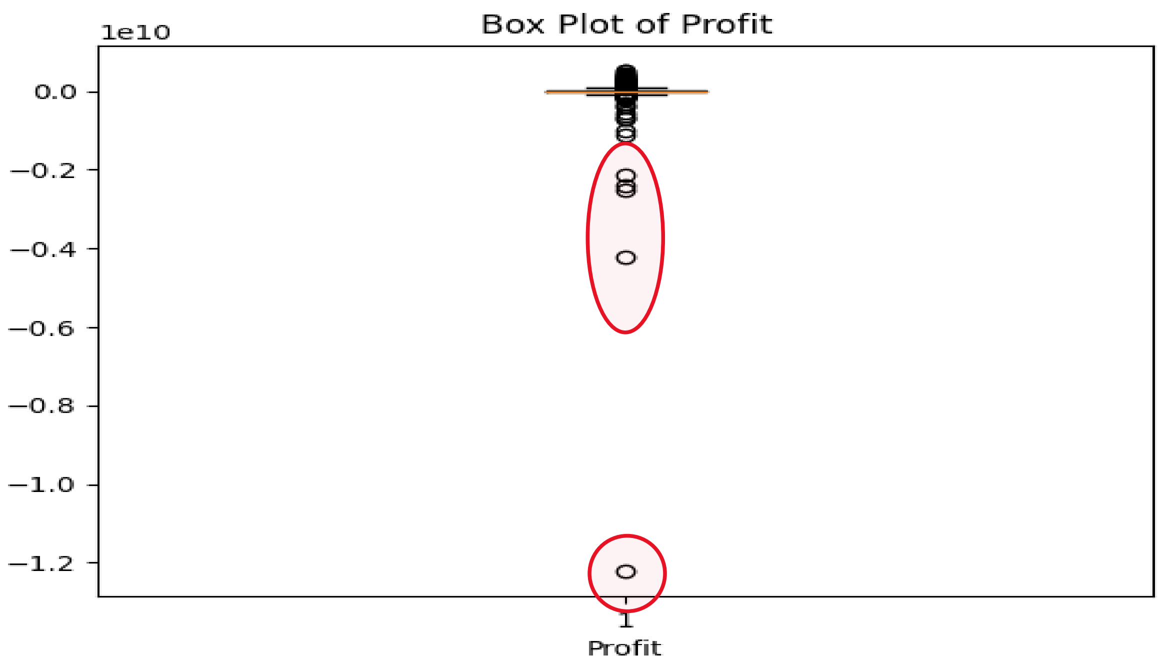
country	content_rating	budget	imdb_score	aspect_ratio	movie_facebook_likes	Profit
USA	PG-13	2.370000e+08	7.9	1.78	33000	5.235058e+08
USA	PG-13	1.500000e+08	7.0	2.00	150000	5.021773e+08
USA	PG-13	2.000000e+08	7.7	2.35	26000	4.586723e+08
USA	PG	1.100000e+07	8.7	2.35	33000	4.499357e+08
USA	PG	1.050000e+07	7.9	1.85	34000	4.244495e+08
...	...	...	...	...	...	...
Japan	PG-13	2.127520e+09	6.9	1.85	973	-2.127110e+09
Japan	PG-13	2.400000e+09	8.4	1.85	11000	-2.397702e+09
Hungary	R	2.500000e+09	7.1	2.35	607	-2.499804e+09
South Korea	R	4.200000e+09	7.7	2.35	4000	-4.199788e+09
South Korea	R	1.221550e+10	7.0	1.85	7000	-1.221330e+10

Plotting Profit vs budget:

```
plt.scatter(profit_data['budget'], profit_data['Profit'])
plt.xlabel('Budget')
plt.ylabel('Profit')
plt.title('Profit vs Budget')
plt.show()
```




```
plt.boxplot(profit_data['Profit'])
plt.xlabel('Profit')
plt.title('Box Plot of Profit')
plt.show()
```



There are 5 outliers present which are the bottom 5 movies in the dataframe sorted by profit.

```
profit_data.sort_values("Profit",ascending=False).tail()
```

movie_title	Profit
Steamboy	-2.127110e+09
Princess Mononoke	-2.397702e+09
Fateless	-2.499804e+09
Lady Vengeance	-4.199788e+09
The Host	-1.221330e+10

Task 3: Top 250 movies

```
df = df[df['num_voted_users'] > 25000]
df=df.sort_values("imdb_score",ascending=False)
df['IMDb_Top_250'] = (df['imdb_score'].rank(ascending=False, method='first') <= 250).astype(int)
df['Rank'] = df['imdb_score'].rank(ascending=False, method='first').astype(int)
df.head(253)
```

movie_title	imdb_score	aspect_ratio	IMDb_Top_250	Rank
The Shawshank Redemption	9.3	1.85	1	1
The Godfather	9.2	1.85	1	2
Fargo	9.0	1.78	1	3
The Dark Knight	9.0	2.35	1	4
The Godfather: Part II	9.0	1.85	1	5
...	...	...	...	...
The Hustler	8.0	2.35	1	249
District 9	8.0	1.85	1	250
Boyhood	8.0	1.85	0	251
Dallas Buyers Club	8.0	2.35	0	252
A Fistful of Dollars	8.0	2.35	0	253

The top 250 movies have been ranked here and they have beenmarked as 1 as they are part of the top 250 in the IMDb_Top_250.

Top Movies not in the english language:

```
Top_Foreign_Lang_Film = dftop_250['language']!="English"]
Top_Foreign_Lang_Film[['Rank', 'movie_title', 'imdb_score', 'language']]
```

	Rank	movie_title	imdb_score	language
4498	6	The Good, the Bad and the Ugly	8.9	Italian
4747	23	Seven Samurai	8.7	Japanese
4029	25	City of God	8.7	Portuguese
2373	27	Spirited Away	8.6	Japanese
3870	42	Airlift	8.5	Hindi
4259	55	The Lives of Others	8.5	German
4921	60	Children of Heaven	8.5	Persian
4105	64	Oldboy	8.4	Korean
4659	73	A Separation	8.4	Persian
3685	75	Rang De Basanti	8.4	Hindi
2970	77	Das Boot	8.4	German
2323	79	Princess Mononoke	8.4	Japanese
1329	82	Baahubali: The Beginning	8.4	Telugu
1298	83	Amélie	8.4	French
2829	87	Downfall	8.3	German
4033	99	The Hunt	8.3	Danish
2734	100	Metropolis	8.3	German
2551	118	Pan's Labyrinth	8.2	Spanish
3550	119	Incendies	8.2	French
2047	128	Howl's Moving Castle	8.2	Japanese
4000	130	The Secret in Their Eyes	8.2	Spanish
4160	146	Lage Raho Munna Bhai	8.2	Hindi
1061	157	Solaris	8.1	Russian
4461	161	The Celebration	8.1	Danish
3553	178	Elite Squad	8.1	Portuguese
2830	181	The Sea Inside	8.1	Spanish
3423	191	Akira	8.1	Japanese
2914	192	Tae Guk Gi: The Brotherhood of War	8.1	Korean
4267	195	Amores Perros	8.1	Spanish
2802	206	The Diving Bell and the Butterfly	8.0	French
3344	221	My Name Is Khan	8.0	Hindi

Task 4: Best Directors

```
dirdf=data.copy()
director_scores = dirdf.groupby('director_name')['imdb_score'].mean()
.reset_index()
director_scores = director_scores.sort_values(by=['imdb_score', 'dire
ctor_name'], ascending=[False, True])
top10directors = director_scores.head(10)['director_name'].tolist()
dirdf['top10director'] = dirdf['director_name'].apply(lambda x: 1 if
x in top10directors else 0)
dirdf.sort_values(['top10director','imdb_score','director_name'],asce
nding=[False,False,True]).head(10)
director_scores.head(10)
```

Here are the top 10 best directors:

	director_name	imdb_score
1083	John Blanchard	9.5
299	Cary Bell	8.7
1619	Mitchell Altieri	8.7
2011	Sadyk Sher-Niyaz	8.7
315	Charles Chaplin	8.6
1605	Mike Mayhall	8.6
428	Damien Chazelle	8.5
1416	Majid Majidi	8.5
1835	Raja Menon	8.5
1979	Ron Fricke	8.5

Task 5 : Popular Genres

Popular genres according movies inthe top 250 created by the top 10 directors.

```
top250movies = df.head(250)
top10directors = director_scores['director_name'].head(10).tolist()
topmovies = top250movies[top250movies['director_name'].isin(top10dire
ctors)]
genres1 = topmovies['genres'].str.split('|', expand=True).stack().res
et_index(level=1, drop=True)
popular_genres1 = genres1.value_counts().sort_values(ascending=False)
popular_genres1
```

```
Drama       4
Family      2
Comedy      1
Action      1
History     1
Thriller    1
War         1
Music       1
dtype: int64
```

Genres ranked according to the top 250 movies:

```
genres2 = top250movies['genres'].str.split('|', expand=True).stack().reset_index(level=1, drop=True)
popular_genres2 = genres2.value_counts().sort_values(ascending=False)
popular_genres2
```

```
Drama          173
Adventure       66
Thriller        56
Crime           51
Action          50
Comedy          41
Sci-Fi          40
Romance         35
Biography       30
Fantasy         30
Mystery         28
Family          24
Animation       20
History         18
Sport            9
Horror           8
Western          8
Musical          7
Documentary      5
Music            3
Film-Noir        1
dtype: int64
```

Task 6

A) 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. Use only the actor_1_name column for extraction. Also, make sure that you use the names 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' for the said extraction.

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

```
actdata=data.copy()
for index,i in actdata.iterrows():
  if i['actor_1_name'] in ["Brad Pitt","Leonardo DiCaprio","Meryl Ste
ep"]:
    print(i['movie_title']+"--- "+i['actor_1_name'])
```

The 3 actors have worked in all these movies:

```
Titanic --- Leonardo DiCaprio
The Great Gatsby --- Leonardo DiCaprio
Inception --- Leonardo DiCaprio
The Curious Case of Benjamin Button --- Brad Pitt
Troy --- Brad Pitt
The Revenant --- Leonardo DiCaprio
Ocean's Twelve --- Brad Pitt
Mr. & Mrs. Smith --- Brad Pitt
The Aviator --- Leonardo DiCaprio
Django Unchained --- Leonardo DiCaprio
Blood Diamond --- Leonardo DiCaprio
The Wolf of Wall Street --- Leonardo DiCaprio
Gangs of New York --- Leonardo DiCaprio
The Departed --- Leonardo DiCaprio
Spy Game --- Brad Pitt
Ocean's Eleven --- Brad Pitt
Shutter Island --- Leonardo DiCaprio
Fury --- Brad Pitt
Seven Years in Tibet --- Brad Pitt
Body of Lies --- Leonardo DiCaprio
Fight Club --- Brad Pitt
Sinbad: Legend of the Seven Seas --- Brad Pitt
Catch Me If You Can --- Leonardo DiCaprio
Interview with the Vampire: The Vampire Chronicles --- Brad Pitt
The Beach --- Leonardo DiCaprio
Revolutionary Road --- Leonardo DiCaprio
The Man in the Iron Mask --- Leonardo DiCaprio
J. Edgar --- Leonardo DiCaprio
The Tree of Life --- Brad Pitt
The Quick and the Dead --- Leonardo DiCaprio
The Assassination of Jesse James by the Coward Robert Ford --- Brad Pitt
Marvin's Room --- Leonardo DiCaprio
Babel --- Brad Pitt
By the Sea --- Brad Pitt
Killing Them Softly --- Brad Pitt
Romeo + Juliet --- Leonardo DiCaprio
True Romance --- Brad Pitt
The Great Gatsby --- Leonardo DiCaprio
Johnny Suede --- Brad Pitt
```

```python
# create new columns for each actor and extract the movies where they were the lead actor
actdata['Meryl_Streep'] = actdata.apply(lambda row: row['movie_title'] if 'Meryl Streep' in row['actor_1_name'] else '', axis=1)
actdata['Leo_Caprio'] = actdata.apply(lambda row: row['movie_title'] if 'Leonardo DiCaprio' in row['actor_1_name'] else '', axis=1)
actdata['Brad_Pitt'] = actdata.apply(lambda row: row['movie_title'] if 'Brad Pitt' in row['actor_1_name'] else '', axis=1)
```

```python
actdata['Combined'] = actdata.apply(lambda row: row if (('Meryl Streep' in row['actor_1_name']) or ('Leonardo DiCaprio' in row['actor_1_name']) or ('Brad Pitt' in row['actor_1_name'])) else '', axis=1)
```

Here are the Brad_pitt and Combined columns for movies in which Brad Pitt starred:

Similar outputs can also be viewed for the other 2 actors.

index	actor_1_name	Brad_Pitt	Combined
101	Brad Pitt	The Curious Case of Benjamin Button	color Color director_name David Fincher num_critic_for_reviews 362.0 actor_2_name Jason Flemyng genres Drama|Fantasy|Romance actor_1_name Brad Pitt movie_title The Curious Case of Benjamin Button num_voted_users 459346 actor_3_name Julia Ormond num_user_for_reviews 822.0 country USA content_rating PG-13 budget 150000000.0 title_year 2008.0 imdb_score 7.8 aspect_ratio 2.35 Meryl_Streep Leo_Caprio Brad_Pitt The Curious Case of Benjamin Button Name: 101, dtype: object
147	Brad Pitt	Troy	color Color director_name Wolfgang Petersen num_critic_for_reviews 220.0 actor_2_name Orlando Bloom genres Adventure actor_1_name Brad Pitt movie_title Troy num_voted_users 381672 actor_3_name Julian Glover num_user_for_reviews 1694.0 country USA content_rating R budget 175000000.0 title_year 2004.0 imdb_score 7.2 aspect_ratio 2.35 Meryl_Streep Leo_Caprio Brad_Pitt Troy Name: 147, dtype: object
254	Brad Pitt	Ocean's Twelve	color Color director_name Steven Soderbergh num_critic_for_reviews 198.0 actor_2_name Julia Roberts genres Crime|Thriller actor_1_name Brad Pitt movie_title Ocean's Twelve num_voted_users 284852 actor_3_name Mini Anden num_user_for_reviews 627.0 country USA content_rating PG-13 budget 110000000.0 title_year 2004.0 imdb_score 6.4 aspect_ratio 2.35 Meryl_Streep Leo_Caprio Brad_Pitt Ocean's Twelve Name: 254, dtype: object
255	Brad Pitt	Mr. & Mrs. Smith	color Color director_name Doug Liman num_critic_for_reviews 233.0 actor_2_name Angelina Jolie Pitt genres Action|Comedy|Crime|Romance|Thriller actor_1_name Brad Pitt movie_title Mr. & Mrs. Smith num_voted_users 348861 actor_3_name Stephanie March num_user_for_reviews 798.0 country USA content_rating PG-13 budget 120000000.0 title_year 2005.0 imdb_score 6.5 aspect_ratio 2.35 Meryl_Streep Leo_Caprio Brad_Pitt Mr. & Mrs. Smith Name: 255, dtype: object
382	Brad Pitt	Spy Game	color Color director_name Tony Scott num_critic_for_reviews 142.0 actor_2_name Stephen Dillane genres Action|Crime|Thriller actor_1_name Brad Pitt movie_title Spy Game num_voted_users 121259 actor_3_name Catherine McCormack num_user_for_reviews 361.0 country Germany content_rating R budget 92000000.0 title_year 2001.0 imdb_score 7.0 aspect_ratio 2.35 Meryl_Streep Leo_Caprio Brad_Pitt Spy Game Name: 382, dtype: object
400	Brad Pitt	Ocean's Eleven	color Color director_name Steven Soderbergh num_critic_for_reviews 186.0 actor_2_name Bernie Mac genres Crime|Thriller actor_1_name Brad Pitt movie_title Ocean's Eleven num_voted_users 402645 actor_3_name Elliott Gould num_user_for_reviews 845.0 country USA content_rating PG-13 budget 85000000.0 title_year 2001.0 imdb_score 7.8 aspect_ratio 2.35 Meryl_Streep Leo_Caprio Brad_Pitt Ocean's Eleven Name: 400, dtype: object
470	Brad Pitt	Fury	color Color director_name David Ayer num_critic_for_reviews 406.0 actor_2_name Logan Lerman genres Action|Drama|War actor_1_name Brad Pitt movie_title Fury num_voted_users 303185 actor_3_name Jim Parrack num_user_for_reviews 701.0 country USA content_rating R budget 68000000.0 title_year 2014.0 imdb_score 7.6 aspect_ratio 2.35 Meryl_Streep Leo_Caprio Brad_Pitt Fury Name: 470, dtype: object
611	Brad Pitt	Seven Years in Tibet	color Color director_name Jean-Jacques Annaud num_critic_for_reviews 76.0 actor_2_name Mako genres Adventure|Biography|Drama|History|War actor_1_name Brad Pitt movie_title Seven Years in Tibet num_voted_users 96385 actor_3_name Victor Wong num_user_for_reviews 119.0 country USA content_rating PG-13 budget 70000000.0 title_year 1997.0 imdb_score 7.0 aspect_ratio 2.35 Meryl_Streep Leo_Caprio Brad_Pitt Seven Years in Tibet Name: 611, dtype: object

B) Find the critic favourite and audience favourite actors.

```
actor_group = data.groupby('actor_1_name')

actor_mean = actor_group[['num_critic_for_reviews', 'num_user_for_reviews']].mean()
actor_mean
```

Here are the actors with their critic and user review ratings:

actor_1_name	num_critic_for_reviews	num_user_for_reviews
50 Cent	98.000000	284.000000
A.J. Buckley	298.000000	345.000000
Aaliyah	137.000000	695.000000
Aasif Mandvi	210.000000	147.000000
Abbie Cornish	270.333333	184.666667
...	...	...
Zoë Kravitz	114.666667	93.666667
Zuhair Haddad	5.000000	1.000000
Álex Angulo	9.000000	7.000000
Ólafur Darri Ólafsson	16.000000	19.000000
Óscar Jaenada	186.000000	139.000000

```
critic_df = actor_mean.sort_values('num_critic_for_reviews', ascending=False)
print(critic_df["num_critic_for_reviews"])
```

Here are the top actors ranked according to critic reviews:

```
Phaldut Sharma          738.0
Peter Capaldi           654.0
Craig Stark             596.0
Bérénice Bejo           576.0
Suraj Sharma            552.0
                        ...
Mike Stanley              1.0
Mike Beckingham           1.0
Marcello Mastroianni      1.0
Manny Perez               1.0
Carrie Bradstreet         1.0
```

```
audience_df = actor_mean.sort_values('num_user_for_reviews', ascending=False)
audience_df["num_user_for_reviews"]
```

Here are the actors ranked according to user reviews:

```
Heather Donahue          3400.0
Christo Jivkov           2814.0
Steve Bastoni            2789.0
Phaldut Sharma           1885.0
Keir Dullea              1736.0
                          ...
Jon Brion                   1.0
Patrick O'Donnell           1.0
Mary Kate Wiles             1.0
Paul Hickert                1.0
Claire Gordon-Harper        1.0
```

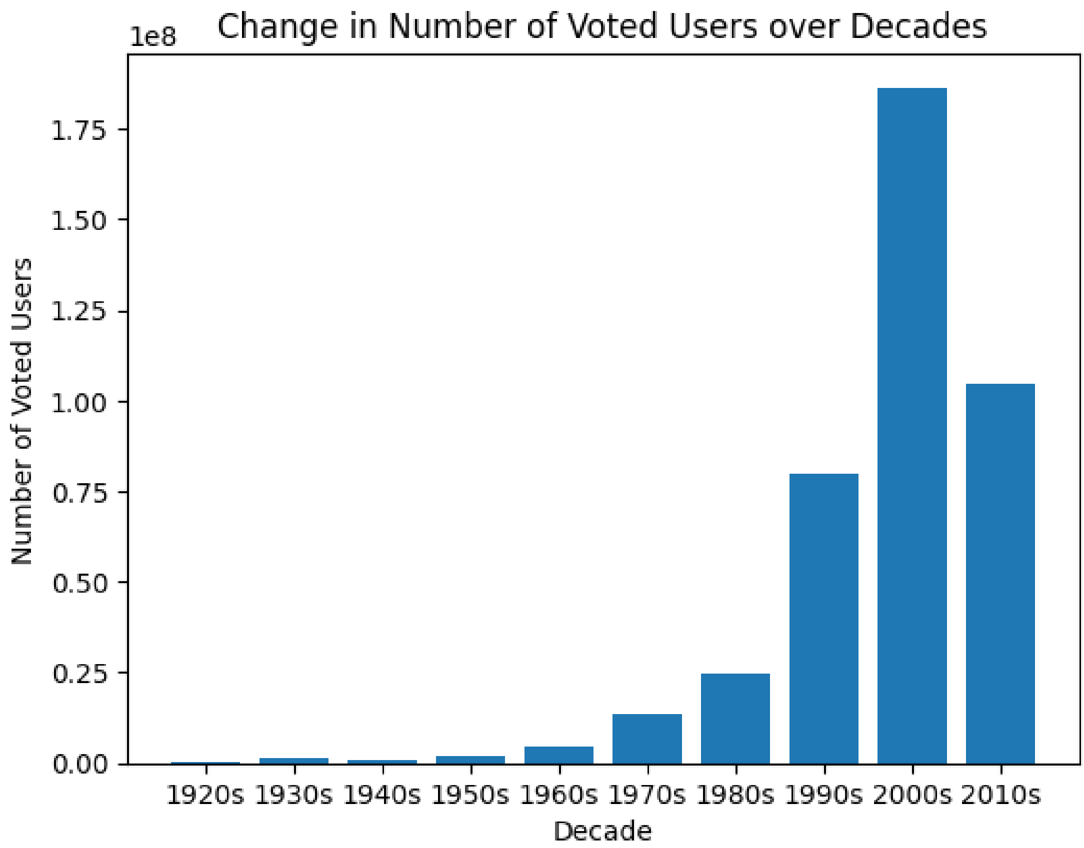
C) Change in number of voted users over decades using barplot.

```
chartdata=data.copy()
bins = [1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000, 2010, 2020]
labels = ['1920s', '1930s', '1940s', '1950s', '1960s', '1970s', '1980s', '1990s', '2000s', '2010s']
chartdata['decade'] = pd.cut(chartdata['title_year'], bins=bins, labels=labels)
df_by_decade = chartdata.groupby('decade')['num_voted_users'].sum().reset_index()
df_by_decade
```

Here are the number of votes cumulated every decade:

	decade	num_voted_users
0	1920s	132420
1	1930s	1233065
2	1940s	962634
3	1950s	2175102
4	1960s	4819970
5	1970s	13740773
6	1980s	24616391
7	1990s	80028936
8	2000s	186323739
9	2010s	104763014

```
plt.bar(df_by_decade['decade'], df_by_decade['num_voted_users'])
plt.xlabel('Decade')
plt.ylabel('Number of Voted Users')
plt.title('Change in Number of Voted Users over Decades')
plt.show()
```



Summary

This project was about IMDB Movie Analysis for which I used Python and Google Colab. I revised my Python skills and how to perform EDA thouroughly.

Thank You-

Devansh Mathur