IMDB MOVIE ANALYSIS

PROJECT DESCRIPTION

This project involves analyzing movies based on certain factors. First of all, I am going to clean the raw data to remove any noise from the data so that no errors come up during the analysis. Then, I am going to visualize the data and create charts to make things clearer.

I will be finding out how movies are earning and going into loss and identifying the key factors that define things clearly in the data. Specifically, I will be identifying which language, director, and actors are most loved by people and whose movies people love to watch. Additionally, I will be looking at which creators make good films and which actors perform great roles.

Finally, I will provide insights on how users can save time and effort in finding the best movies to watch in theaters, how directors can see which movies people like the most, and how OTT platforms can make money from films, engage users on their platforms, increase screen time, and ultimately grow more in the future.

PROJECT APPROACH

This project was developed using Python in Jupyter Notebook. Python was chosen for its ease of data analysis, as well as its ability to handle large amounts of data without overloading the processor. To begin, the necessary libraries such as pandas and numpy were imported. These libraries are used for data visualization. Using pandas, the file location containing the data was specified and the data was read into the notebook. Then, a data frame was created for the specific file and analyzed to gain insights into the data.

TECK STACK USED

Python is a high-level, interpreted programming language that was first released in 1991 by Guido van Rossum. It is known for its simplicity, ease of use, and readability. It is free to use, distribute, and modify.

Python has a wide range of applications, including web development, data analysis, scientific computing, artificial intelligence, machine learning, and automation

Python is one of the programming languages that can be used in Jupyter Notebook. This means that users can write Python code in a Jupyter Notebook cell, run the code, and see the output immediately in the notebook.

RESULT AND INSIGHTS

- After downloading the CSV file, I used the Pandas and NumPy libraries to read the data from the CSV file and convert it into a DataFrame by using the following commands:

```
import pandas as pd
import numpy as np
df=pd.read_csv("F:\\IMDB_Movies.csv")
```

1. **Cleaning the data::** This is one of the most important step to perform before moving forward with the analysis. Use your knowledge learned till now to do this. (Dropping columns, removing null values, etc.)
   Your task: Clean the data.

- Cleaning is one of the most important parts of data analysis. Without proper cleaning, we cannot proceed with the analysis as it may result in errors that can affect the accuracy of our results. Therefore, it is crucial to clean the data before conducting any analysis to ensure that the data is accurate, consistent, and reliable.

- After seeing the DataFrame and task given, I have identified that the following **columns** are not needed for the analysis and can be removed for cleaning the data:

```
df=df.drop(['color','duration','director_facebook_likes','cast_total_facebook_likes','facenumber_in_poster','plot_keywords','movie_imdb_link','country','content_rating','aspect_ratio','movie_facebook_likes','actor_3_facebook_likes','actor_2_facebook_likes','actor_1_facebook_likes','actor_3_name','actor_2_name'],axis=1)
```

- Removing unnecessary **columns** can improve the performance of data analysis and reduce the risk of errors. However, it's important to ensure that the removed columns are not required for future analysis or reporting.

- Since we are analyzing raw data, it is likely that there will be some noise in the data. To check for noise in the data, we can use the following command:

  - **df**.isnull().sum()
  - **df** = df.dropna()

- After checking the data types of the columns, I identified that one column has an incorrect data type, and I converted it to the appropriate data type. This is an important step in data cleaning to ensure that the data is consistent and accurate for further analysis.

  - **df**.dtypes
  - **df**['num_user_for_reviews']=df['num_user_for_reviews'].astype (float)

- After removing the duplicate values from the table, I have further improved it to get rid of errors. This step is important to ensure that the data is free from any redundant or irrelevant information that can affect the accuracy of our analysis. By removing duplicates and improving the data, we can obtain a more reliable and accurate dataset for analysis.

  df = df.drop_duplicates()

- After cleaning and improving the dataset, I reset the index to make it more formalized and attractive. This step is important for presenting the data in a more organized and easy-to-read format. By resetting the index, we can ensure that the data is presented in a consistent and structured way, which can help in better understanding and interpretation of the data.

- **df**.reset_index(inplace=True)
- **df**.drop(['index'], axis=1, inplace=True)

- Great! Now that we have cleaned and prepared our data by removing noise, duplicates, and discrepancies, we can proceed with our analysis. This will help us to gain insights, make informed decisions, and draw conclusions based on the data. It is important to ensure that our data is accurate and reliable before conducting any analysis to avoid making erroneous conclusions.

2. **Movies with highest profit:** Create a new column called profit which contains the difference of the two columns: gross and budget. Sort the column using the profit column as reference. Plot profit (y-axis) vs budget (x- axis) and observe the outliers using the appropriate chart type.
   Your task: Find the movies with the highest profit?

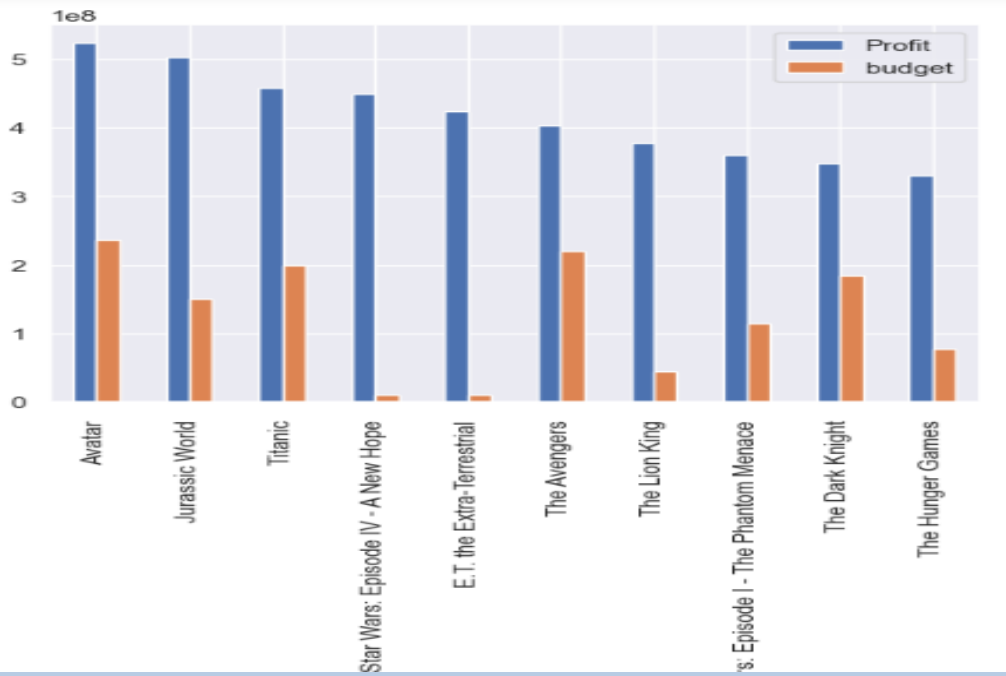
- df['Profit']=df['gross']-df['budget']
- **Maximum_Profit_Movies** = df.nlargest(10, 'Profit')[['movie_title', 'Profit']].reset_index()
- **Maximum_Profit_Movies**.drop(['index'], axis=1, inplace=True)

	movie_title	Profit
0	Avatar	523505847.0
1	Jurassic World	502177271.0
2	Titanic	458672302.0
3	Star Wars: Episode IV - A New Hope	449935665.0
4	E.T. the Extra-Terrestrial	424449459.0
5	The Avengers	403279547.0
6	The Lion King	377783777.0
7	Star Wars: Episode I - The Phantom Menace	359544677.0
8	The Dark Knight	348316061.0
9	The Hunger Games	329999255.0

- These are the top 10 movies that most people have watched, leading to maximum profits and fame for the movie, the platform, and the producer. These movies have also received the highest ratings on IMDb, and are recommended across various platforms as people love to watch them.

CHART

- **Maximum_Profit_Movies_Chart**=df.nlargest(10,'Profit')[['movie_title', 'budget', 'Profit']].reset_index()
- **bar_chart** = Maximum_Profit_Movies_Chart[['movie_title', 'Profit', 'budget']].plot(kind='bar', x='movie_title')



- This graph depicts an indirect relationship between a movie's budget and the profit earned. It is not always the case that higher budget films will result in higher profits.

- Others may require a higher budget to achieve the same level of profitability. This is due to a variety of factors such as the quality of the script, the popularity of the actors, and the effectiveness of the marketing campaign.

- It is important to note that while budget is an important factor in the success of a movie, it is not the only one. Other factors such as the genre, the target audience, and the timing of the release can also play a significant role in determining a film's profitability. Therefore, it is essential to consider a range of factors when making decisions about film production and investment.

3. **Top 250:** Create a new column IMDb_Top_250 and store the top 250 movies with the highest IMDb Rating (corresponding to the column: imdb_score). Also make sure that for all of these movies, the num_voted_users is greater than 25,000. Also add a Rank column containing the values 1 to 250 indicating the ranks of the corresponding films.

   Extract 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. You can use your own imagination also!
   Your task: Find IMDB Top 250

- **IMDb =** df[(df['num_voted_users']>25000) & (df['language']=='English')]
- **Top_Movies_250**=IMDb.sort_values(by=['imdb_score'],ascending=[False])[['movie_title','imdb_score','language']].head(250).reset_index()
- **Top_Movies_250**.drop(['index'], axis=1, inplace=True)
- **Top_Movies_250**.reset_index()
- **Top_Movies_250** = Top_Movies.reset_index(drop=False).rename(columns={'index': 'Rank','movie_title': 'IMDb_Top_250'})

	Rank	Rank	IMDb_Top_250	imdb_score	language
0	0	0	The Shawshank Redemption	9.3	English
1	1	1	The Godfather	9.2	English
2	2	2	The Dark Knight	9.0	English
3	3	3	The Godfather: Part II	9.0	English
4	4	4	Schindler's List	8.9	English
...	...	...	...	...	...
245	245	245	Ocean's Eleven	7.8	English
246	246	246	Changeling	7.8	English
247	247	247	Airplane!	7.8	English
248	248	248	Blazing Saddles	7.8	English
249	249	249	Fantastic Mr. Fox	7.8	English

250 rows × 5 columns

- **Foreign_language =** df[(df['num_voted_users']>25000) & (df['language']!='English')]
- **Foreign_language**=Foreign_language.sort_values(by=['imdb_score'],ascending=[False])[['movie_title','imdb_score','language']].head(250).reset_index()
- **Foreign_language**.drop(['index'], axis=1, inplace=True)
- **Foreign_language**.reset_index()
- **Foreign_language** =Foreign_language.reset_index(drop=False).rename(columns={'index': 'Rank','movie_title': 'Top_Foreign_Lang_Film'})

	Rank	Top_Foreign_Lang_Film	imdb_score	language
0	0	The Good, the Bad and the Ugly	8.9	Italian
1	1	City of God	8.7	Portuguese
2	2	Seven Samurai	8.7	Japanese
3	3	Spirited Away	8.6	Japanese
4	4	Children of Heaven	8.5	Persian
...	...	...	...	...
86	86	Night Watch	6.5	Russian
87	87	The Interpreter	6.4	Aboriginal
88	88	Dead Snow	6.4	Norwegian
89	89	The Legend of Zorro	5.9	Spanish
90	90	In the Land of Blood and Honey	4.3	Bosnian

91 rows × 4 columns

- Above are the top movies which have the highest ratings and are more popular among users. These ratings help users to see which movie is the best, and they can watch it without thinking whether the film is good or not, saving their time and effort in finding a popular and good movie. This feature attracts customers to the website, and it is working for almost all the users who find it challenging to search for a good movie. These movies have everything that people want in a movie.

4. **Best Directors:** Group the column using the director_name column.

   Find out the top 10 directors for whom the mean of imdb_score is the highest and store them in a new column top10director. In case of a tie in IMDb score between two directors, sort them alphabetically.
   Your task: Find the best directors.

- **Top_Director**=df.groupby('director_name')['imdb_score'].mean().to_frame().sort_values(by=['imdb_score','director_name'],ascending=[False,True]).head(10).reset_index()
- **Top_Director** =Top_Director.rename(columns={'director_name': 'Top 10 Director','imdb_score':'IMDB_Score_Mean'})

	Top 10 Director	IMDB_Score_Mean
0	Charles Chaplin	8.600000
1	Tony Kaye	8.600000
2	Alfred Hitchcock	8.500000
3	Damien Chazelle	8.500000
4	Majid Majidi	8.500000
5	Ron Fricke	8.500000
6	Sergio Leone	8.433333
7	Christopher Nolan	8.425000
8	Asghar Farhadi	8.400000
9	Marius A. Markevicius	8.400000

- Here are the top 10 directors whose movies have received the highest ratings from users. This information is beneficial for users, as well as for the platform and the actors/actresses involved.

- The table above can be a factor for actors/actresses to choose their roles based on the director. Since it is the director who makes the movies and knows what kind of films people love, actors/actresses may choose to sign on to movies created by popular directors. This can help

them gain fame, attract more brands, and protect their public image. As people are more attracted to actors/actresses who work with popular directors, it can help them grow and succeed in the future.

- The above table can also help platforms and websites recommend movies to people, encouraging them to watch more entertaining films and increase their screen time. It can also help them launch movies on their platform that are loved by users, as promoting unpopular movies could damage their reputation and result in significant losses.

5. **Popular Genres:** Perform this step using the knowledge gained while performing previous steps.
   Your task: Find popular genres

- **Top_Popularity_Genres**=df.groupby('genres')['movie_title'].count().to_frame()
- **Top_Popularity_Genres**= Top_Popularity_Genres.rename(columns={'movie_title': 'Total_Movies'})
- **Top_Popularity_Genres**.sort_values(['Total_Movies'],ascending=[False]).head(10)

genres	Total_Movies
Drama	153
Comedy|Drama|Romance	151
Comedy|Drama	147
Comedy	145
Comedy|Romance	135
Drama|Romance	119
Crime|Drama|Thriller	82
Action|Crime|Thriller	55
Action|Crime|Drama|Thriller	50
Action|Adventure|Sci-Fi	46

- The top 10 genres are mentioned above, and they are helpful for audiences to identify their preferences and for creators to know what kind of audience they want to appeal to. The most popular genres is Drama. By making films in these genres, creators can attract more audiences and earn more profits from films.

- The website can recommend movies of these genres so that people get engaged in watching movies and also share them with their friends, making the movie more popular among people. The platform can launch these movies in the top movies category, which will not only help creators but also the platform to make profits and avoid losing money after releasing the movie on the platform.

6. **Charts:** 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.
   Group the combined column using the actor_1_name column.
   Find the mean of
   the num_critic_for_reviews and num_users_for_review and identify the actors which have the highest mean.

   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 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 this in a new data frame called df_by_decade.
Your task: Find the critic-favorite and audience-favorite actors

- **Meryl_Streep**=df[df['actor_1_name']=='Meryl Streep']['movie_title'].to_frame()
  Meryl_Streep= Meryl_Streep.rename(columns={'movie_title': 'Meryl Streep'})
- **Leo_Caprio**=df[df['actor_1_name']=='Leonardo DiCaprio']['movie_title'].to_frame()
  Leo_Caprio= Leo_Caprio.rename(columns={'movie_title': 'Leo_Caprio'})
- **Brad_Pitt**=df[df['actor_1_name']=='Brad Pitt']['movie_title'].to_frame()
  Brad_Pitt= Brad_Pitt.rename(columnnnns={'movie_title': 'Brad Pitt'})

	Meryl Streep
0	It's Complicated
1	The River Wild
2	Julie & Julia
3	The Devil Wears Prada
4	Lions for Lambs
5	Out of Africa
6	Hope Springs
7	One True Thing
8	The Hours
9	The Iron Lady
10	A Prairie Home Companion

	Brad Pitt
0	The Curious Case of Benjamin Button
1	Troy
2	Ocean's Twelve
3	Mr. & Mrs. Smith
4	Spy Game
5	Ocean's Eleven
6	Fury
7	Seven Years in Tibet
8	Fight Club
9	Sinbad: Legend of the Seven Seas
10	Interview with the Vampire: The Vampire Chroni...
11	The Tree of Life
12	The Assassination of Jesse James by the Coward...
13	Babel
14	By the Sea
15	Killing Them Softly
16	True Romance

	Leo_Caprio
0	Titanic
1	The Great Gatsby
2	Inception
3	The Revenant
4	The Aviator
5	Django Unchained
6	Blood Diamond
7	The Wolf of Wall Street
8	Gangs of New York
9	The Departed
10	Shutter Island
11	Body of Lies
12	Catch Me If You Can
13	The Beach
14	Revolutionary Road
15	The Man in the Iron Mask
16	J. Edgar
17	The Quick and the Dead
18	Marvin's Room
19	Romeo + Juliet

- **Meryl Streep** has acted in a total of 11 movies.
- **Brad Pitt** has acted in 17 movies.
- **Leonardo DiCaprio** has acted in 20 movies.

➢ **Meryl_Streep**=df[df['actor_1_name']=='Meryl Streep']
Meryl_Streep_mean=Meryl_Streep.groupby('actor_1_name')[['num_critic_for_reviews','num_user_for_reviews']].mean().reset_index()

➢ **Leo_Caprio**=df[df['actor_1_name']=='Leonardo DiCaprio']
Leo_Caprio_mean=Leo_Caprio.groupby('actor_1_name')[['num_critic_for_reviews','num_user_for_reviews']].mean().reset_index()

➢ **Brad_Pitt**=df[df['actor_1_name']=='Brad Pitt']
Brad_Pitt_mean=Brad_Pitt.groupby('actor_1_name')[['num_critic_for_reviews','num_user_for_reviews']].mean().reset_index()

	actor_1_name	num_critic_for_reviews	num_user_for_reviews
0	Leonardo DiCaprio	322.2	922.55

	actor_1_name	num_critic_for_reviews	num_user_for_reviews
0	Brad Pitt	245.0	742.352941

	actor_1_name	num_critic_for_reviews	num_user_for_reviews
0	Meryl Streep	181.454545	297.181818

- **Leonardo DiCaprio is the most favored actor by both the audience and the critics.**

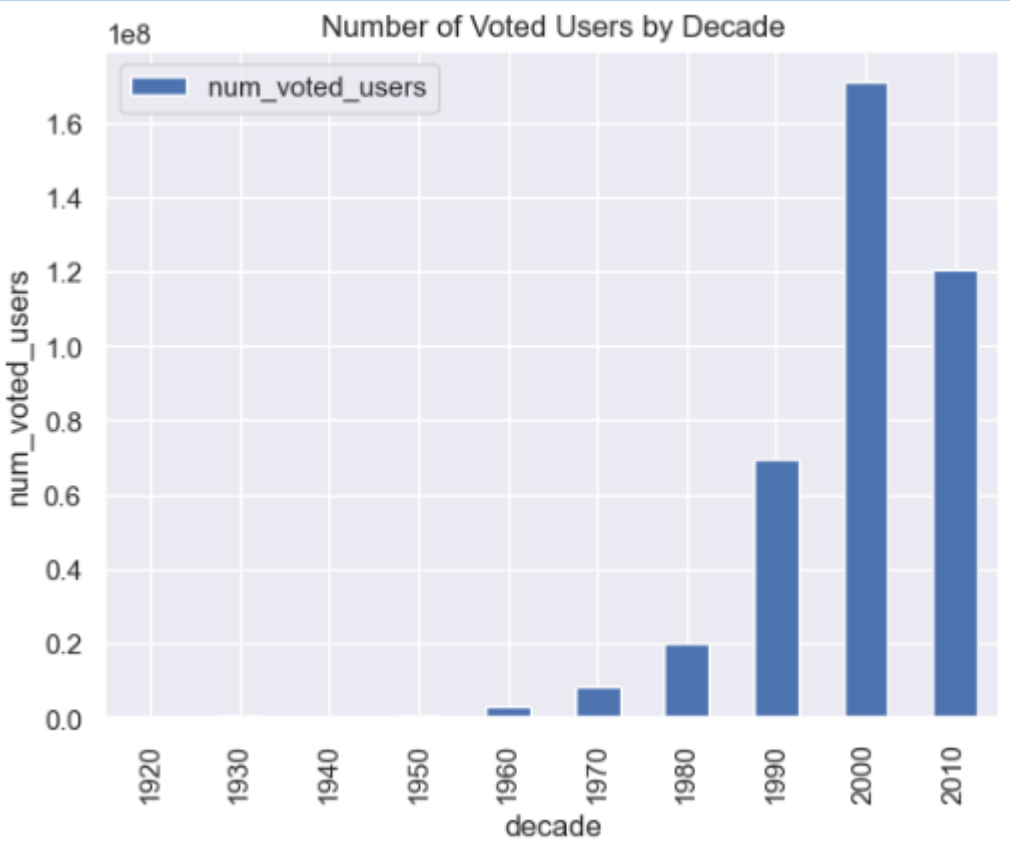
As you can see, among the 3 actors, **Leonardo DiCaprio** has the highest mean of 'num_critic_for_reviews' and 'num_user_for_reviews.' Critics can provide valuable insights into a work of art, helping readers or viewers understand it more deeply, appreciate it more fully, or make informed decisions about whether to engage with it.

CHART

- df[**'decade'**] = (df['title_year'] // 10 * 10).astype(int)
- **df_by_decade** =pd.DataFrame(df.groupby('decade')['num_voted_users'].sum()).reset_index()
- **df_by_decade**.sort_values('decade')

	decade	num_voted_users
0	1920	116387
1	1930	528405
2	1960	947615
3	1970	251079
4	1980	2487213
5	1990	12869196
6	2000	28484675
7	2010	48735658

- **df_by_decade.**plot(kind='bar', x='decade', y='num_voted_users')
- plt.xlabel('decade')
- plt.ylabel('num_voted_users')
- plt.title('Number of Voted Users by Decade')
- plt.show()



- From the above graph, it is evident that the number of users who voted was at its highest in the year 2000. As we have progressed towards the present day, we now have access to feedback systems and voting platforms that allow users to vote and provide feedback on movies.

INSIGHTS

- By analyzing the data, I can say that the movies with the highest IMDb ratings have earned higher profits. These movies are loved most by the audience and are the ones that people want to see.
- Drama are the most popular genres among audiences, so creators should include these in their movies to make them more popular.
- Looking at the graph between budget and profit, I can say that it it indirectly depends on each other.
- Websites and other platforms can suggest good directors with the highest IMDb ratings to make an impression on the audience to watch movies made by those directors. If people are confused about which movie they want to see, they can see more movies from that particular director in the future.
- By seeing the IMDb rating of actors/actresses, creators can choose the best role for them in their movies. It is also beneficial for people to keep engaging with watching movies of talented actors who play good roles, making a good impact on the audience.
- According to all the points, OTT platforms and websites can launch movies according to the audience's choice and take contracts to launch movies on their platform. This is beneficial for both the audience and platform, saving them time and effort. OTT platforms can give suggestions according to the people's choice, recommending movies immediately after the finish of one movie to keep engaging people to watch more movies. This increases the screen time and is good for the platform to earn profit and become popular, earning fame and growing bigger in the future. It will also help more brands approach the platform, directors, or actors/actresses to grow in the future.
- It is also beneficial to see the choices of the youth, and according to that, movies can be made to make a good impression on the audience every time.

RESULT

- To make this project, it takes a lot of time and effort. First of all, I had to make my mind think broader and think about how to approach the project. It also helped me clear my concepts and sharpen my skills in Python.
- Through this project, I learned that directors and platforms make movies according to the audience's needs. By analyzing data, they can attract and engage the audience and make more profits, becoming more popular than other websites and platforms.
- I also learned how directors choose roles for particular actors and actresses in movies, and the same goes for actors/actresses signing up for a role in a movie.
- As a future data analyst, I can help companies grow and make profits by rectifying any mistakes and improving their strategies. This will not only be beneficial for the company but also help me gain valuable experience and skills.

PROJECT LINK

https://drive.google.com/file/d/1ONG6I1rjJiq8x-xaVPK6y7hzNfjJu48p/view?usp=share_link

THANK YOU