Name: Shreya Bhattacharjee
Roll no:6514
Data science

IMDB_DATASET

Pillai College of Arts, Commerce & Science

(Autonomous)
Affiliated to University of Mumbai

NAAC Accredited 'A' grade (3 cycles)
Best College Award by University of Mumbai

ISO 9001:2015 Certified

CERTIFICATE

This is to certify that Mr. /Miss. Shreya Bhattacharjee of S.Y B.Sc. C.S. Semester IV has completed the practical work in the Subject of Data Science during the academic year 2021-22 under the guidance of Prof. _Sanjana Bhangale being the partial requirement for the fulfilment of the curriculum of Degree of Bachelor of Science in Computer Science, University of Mumbai.

Content

- Introduction
- Libraries used
- Code and output
- conclusion
- About project

- The dataset which I have chosen is about cancer detection in this we are having 1000 rows and 16columns.

The dataset we have taken from Kaggle.com

1. NumPy: NumPy can be used to **perform a wide variety of mathematical operations on arrays**. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

2. Pandas : Pandas is a **Python library for data analysis**. Started by Wes McKinney in 2008 out of a need for a powerful and flexible quantitative analysis tool, pandas has grown into one of the most popular Python libraries. It has an extremely active community of contributors.

3. Matplotlib: is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

4. Seaborn: Seaborn is **a data visualization library built on top of matplotlib** and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data. ... Visualizing univariate and bivariate distribution.

CODE

Importing libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

importing csv file

```
from google.colab import files
uploaded=files.upload()
df=pd.read_csv('imdb_top_1000.csv')
df.head(5)
```

Edit View Insert Runtime Tools Help

+ Text

RAM Disk Editing

Saving imdb_top_1000.csv to imdb_top_1000 (3).csv

	Poster_Link	Series_Title	Released_Year	Certificate	Runtime	Genre	IMDB_Rating	Overview	Meta_score	Director
0	https://m.media-amazon.com/images/M/MV5BMDFkYT...	The Shawshank Redemption	1994	A	142 min	Drama	9.3	Two imprisoned men bond over a number of years...	80.0	Frank Darabont
1	https://m.media-amazon.com/images/M/MV5BM2MyNj...	The Godfather	1972	A	175 min	Crime, Drama	9.2	An organized crime dynasty's aging patriarch t...	100.0	Francis Ford Coppola
2	https://m.media-amazon.com/images/M/MV5BMTMxNT...	The Dark Knight	2008	UA	152 min	Action, Crime, Drama	9.0	When the menace known as the Joker wreaks havo...	84.0	Christopher Nolan
3	https://m.media-amazon.com/images/M/MV5BMWMwMG...	The Godfather: Part II	1974	A	202 min	Crime, Drama	9.0	The early life and career of Vito Corleone in ...	90.0	Francis Ford Coppola
4	https://m.media-	12 Angry Men	1957	U	96 min	Crime,	9.0	A jury holdout attempts	96.0	Sidney

```
df.tail(10)
```

995	https://m.media-amazon.com/images/M/MV5BNGEwMT...	Breakfast at Tiffany's	1961	A	115 min	Comedy, Drama, Romance	7.6	A young New York socialite becomes interested ...	76.0
996	https://m.media-amazon.com/images/M/MV5BODk3Yj...	Giant	1956	G	201 min	Drama, Western	7.6	Sprawling epic covering the life of a Texas ca...	84.0
997	https://m.media-amazon.com/images/M/MV5BM2U3Yz...	From Here to Eternity	1953	Passed	118 min	Drama, Romance, War	7.6	In Hawaii in 1941, a private is cruelly punish...	85.0
998	https://m.media-amazon.com/images/M/MV5BZTBmMj...	Lifeboat	1944	NaN	97 min	Drama, War	7.6	Several survivors of a torpedoed merchant ship...	78.0
999	https://m.media-amazon.com/images/M/MV5BMTY5OD...	The 39 Steps	1935	NaN	86 min	Crime, Mystery, Thriller	7.6	A man in London tries to help a counter-espion...	93.0

```
df.shape
```

```
(1000, 16)
```

```
print("The shape of the dataset is: {} rows and {}
```

The shape of the dataset is: 1000 rows and 16 columns

```
df.dtypes
```

```
Poster_Link       object
Series_Title      object
Released_Year     object
Certificate       object
Runtime           object
Genre             object
IMDB_Rating      float64
Overview          object
Meta_score       float64
Director          object
Star1             object
Star2             object
Star3             object
Star4             object
No_of_Votes        int64
Gross             object
dtype: object
```

```
df[df.duplicated()]
```

Poster_Link Series_Title Released_Year Certificate Runtime Genre IMDB_Rating Overview Meta_score Director Star1 Star2 Star3 Star4 No_of_Votes Gross

```
df.describe()
```

	IMDB_Rating	Meta_score	No_of_Votes
count	1000.000000	843.000000	1.000000e+03
mean	7.949300	77.971530	2.736929e+05
std	0.275491	12.376099	3.273727e+05
min	7.600000	28.000000	2.508800e+04
25%	7.700000	70.000000	5.552625e+04
50%	7.900000	79.000000	1.385485e+05
75%	8.100000	87.000000	3.741612e+05
max	9.300000	100.000000	2.343110e+06

```
df.corr()
```

	IMDB_Rating	Meta_score	No_of_Votes
IMDB_Rating	1.000000	0.268531	0.494979
Meta_score	0.268531	1.000000	-0.018507
No_of_Votes	0.494979	-0.018507	1.000000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 16 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Poster_Link    1000 non-null   object
 1   Series_Title   1000 non-null   object
 2   Released_Year  1000 non-null   object
 3   Certificate    899 non-null    object
 4   Runtime        1000 non-null   object
 5   Genre          1000 non-null   object
 6   IMDB_Rating    1000 non-null   float64
 7   Overview       1000 non-null   object
 8   Meta_score     843 non-null    float64
 9   Director       1000 non-null   object
 10  Star1          1000 non-null   object
 11  Star2          1000 non-null   object
 12  Star3          1000 non-null   object
 13  Star4          1000 non-null   object
 14  No_of_Votes    1000 non-null   int64
 15  Gross          831 non-null    object
dtypes: float64(2), int64(1), object(13)
memory usage: 125.1+ KB
```

```
df.isna().sum()
```

```
df.isna().sum()
```

```
Poster_Link         0
Series_Title        0
Released_Year       0
Certificate       101
Runtime             0
Genre               0
IMDB_Rating         0
Overview            0
Meta_score        157
Director            0
Star1               0
Star2               0
Star3               0
Star4               0
No_of_Votes         0
Gross             169
dtype: int64
```

```
df.columns
```

```
Index(['Poster_Link', 'Series_Title', 'Released_Year', 'Certificate',
       'Runtime', 'Genre', 'IMDB_Rating', 'Overview', 'Meta_score', 'Director',
       'Star1', 'Star2', 'Star3', 'Star4', 'No_of_Votes', 'Gross'],
      dtype='object')
```

```
df.dtypes
```

```
Poster_Link        object
Series_Title       object
Released_Year      object
Certificate        object
Runtime            object
Genre              object
IMDB_Rating       float64
Overview           object
Meta_score        float64
Director           object
Star1              object
Star2              object
Star3              object
Star4              object
No_of_Votes         int64
Gross              object
dtype: object
```

```
df.iloc[:,1:12].corr()
```

	IMDB_Rating	Meta_score
IMDB_Rating	1.000000	0.268531
Meta_score	0.268531	1.000000

```
df.mean()
```

```
IMDB_Rating      7.94930
Meta_score      77.97153
No_of_Votes   273692.91100
dtype: float64
```

```
df.median()
```

```
IMDB_Rating      7.9
Meta_score      79.0
No_of_Votes   138548.5
dtype: float64
```

```
df['Genre'].unique()[:5]
```

```
array(['Drama', 'Crime, Drama', 'Action, Crime, Drama',
       'Action, Adventure, Drama', 'Biography, Drama, History'],
      dtype=object)
```

```
df['genre'] = df['Genre'].apply(lambda text: text.split(',')[0])
df.drop(columns='Genre', inplace=True)
df['genre'].value_counts()
```

```
Drama         289
Action        172
Comedy        155
Crime         107
Biography      88
Animation      82
Adventure      72
Mystery        12
Horror         11
Western         4
Film-Noir       3
Family          2
```

```
Fantasy         2
Thriller        1
Name: genre, dtype: int64
```

```
df.Gross = df.Gross.apply(lambda x : str(x).replace(',',''))
```

```
df.Gross = df.Gross.astype

('float64')
```

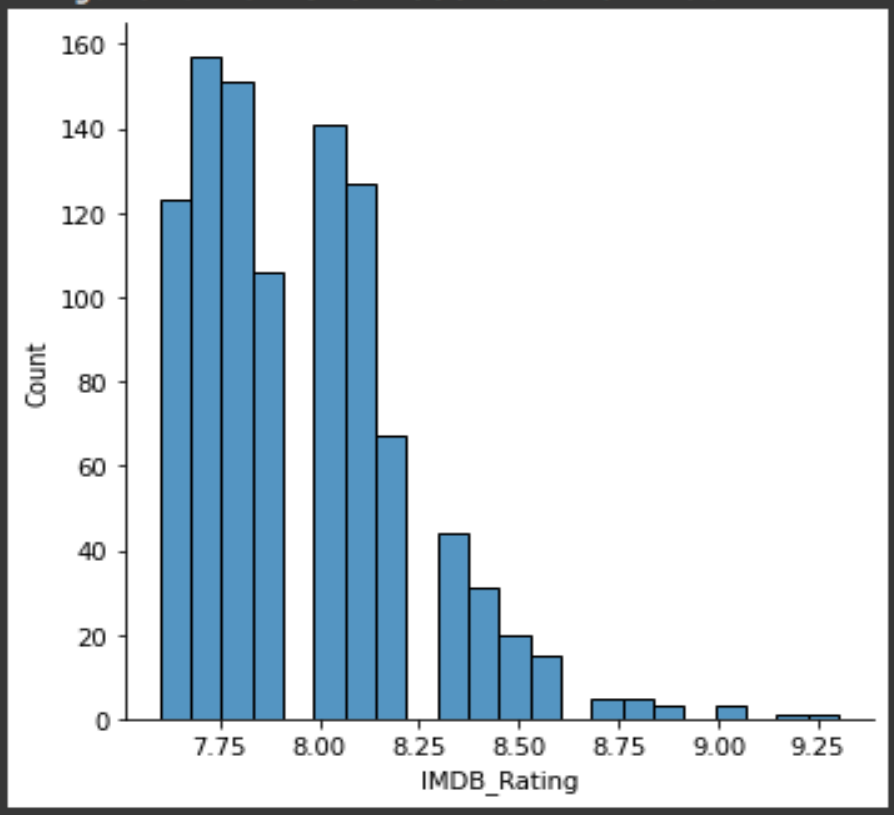
```
df.Gross.dtypes
```

dtype('float64')

Data visualization

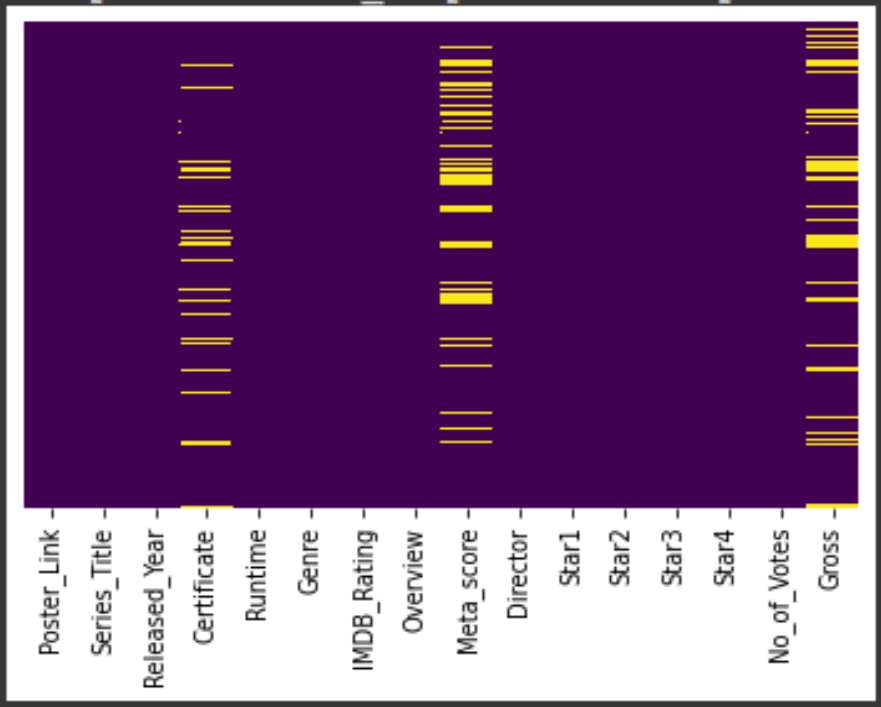
```
plt.figure(figsize = (35,15))

sns.displot(data = df, x = 'IMDB_Rating')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fb5d7cea150>
<Figure size 2520x1080 with 0 Axes>
```




```
sns.heatmap(df.isnull(),yticklabels=False, cbar=False, cmap='viridis')
```

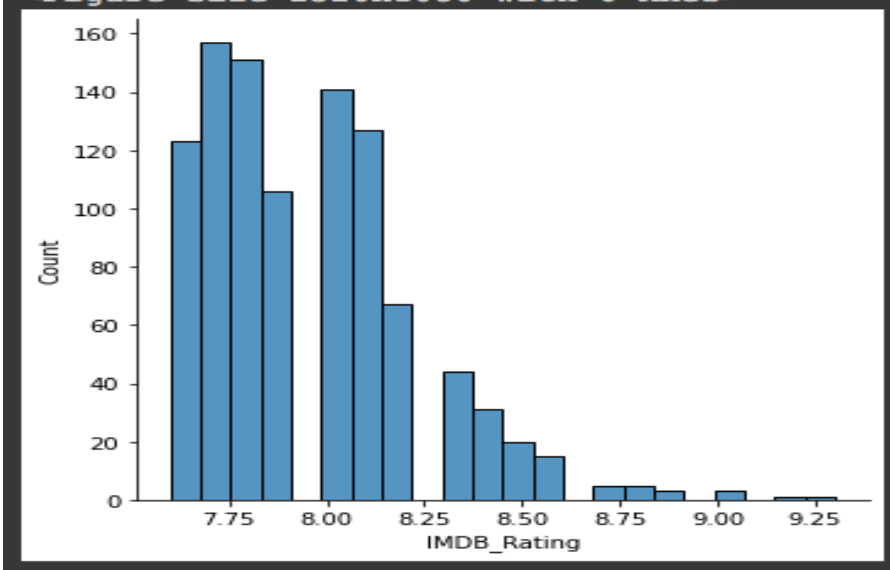
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb5c9272610>
```




```
df.Gross = df.Gross.apply(lambda x : str(x).replace(',',''))
```

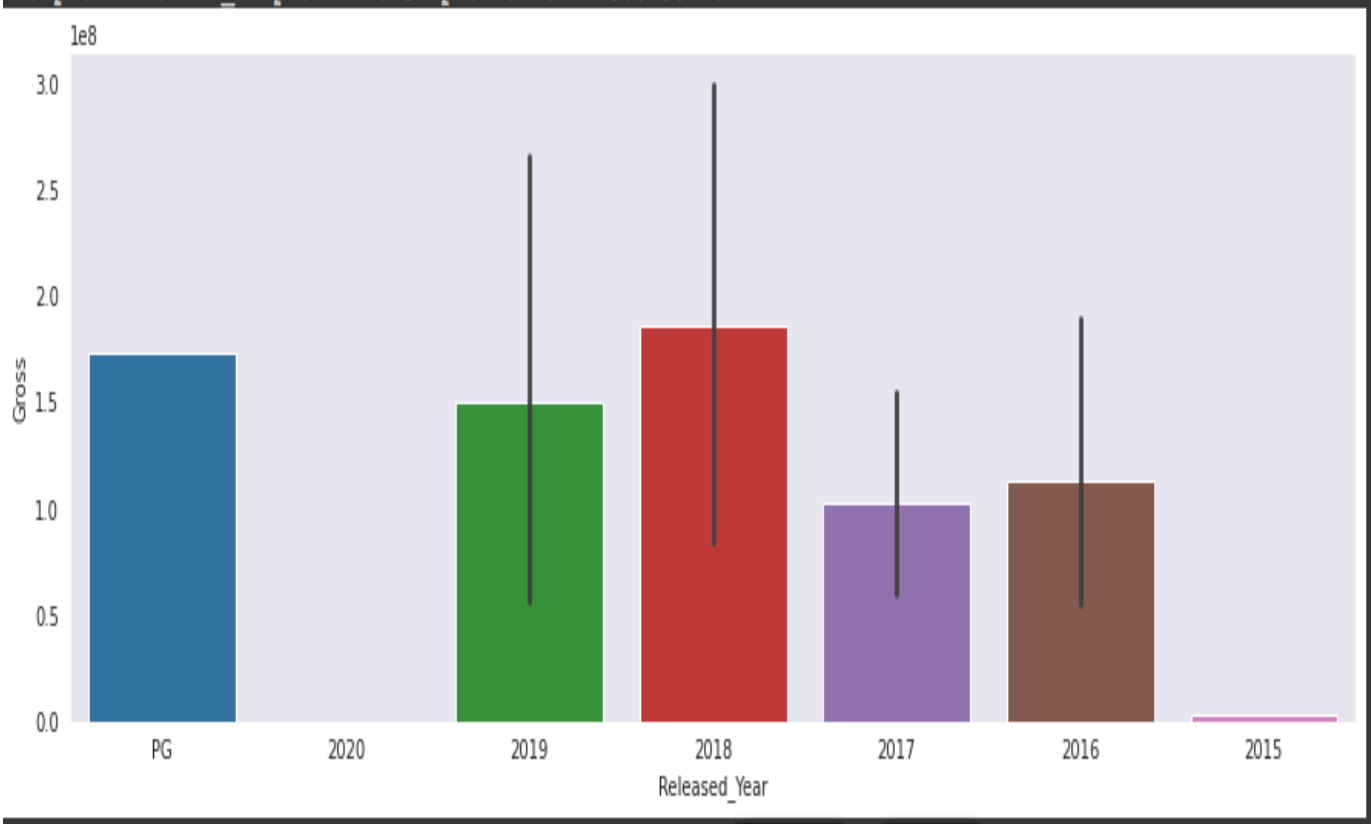
```
plt.figure(figsize = (35,15))

sns.displot(data = df, x = 'IMDB_Rating')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fb5c8c91550>
<Figure size 2520x1080 with 0 Axes>
```




```
sns.set_style('dark')
plt.figure(figsize = (15,5))
sns.barplot(x = 'Released_Year', y = 'Gross', data =
df.sort_values('Released_Year',ascending=False).head(100))
```

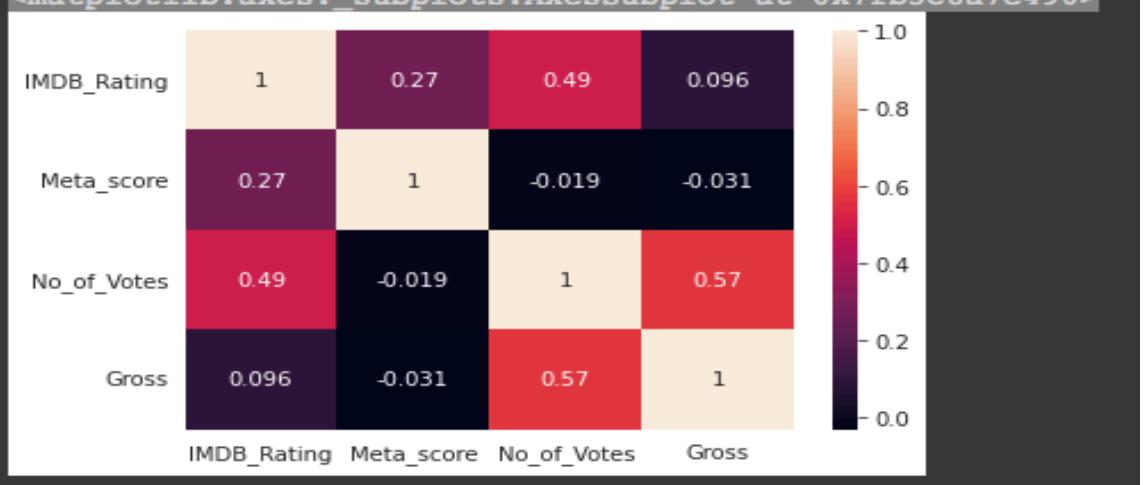
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb5c8430f10>
```



+ Code + Text

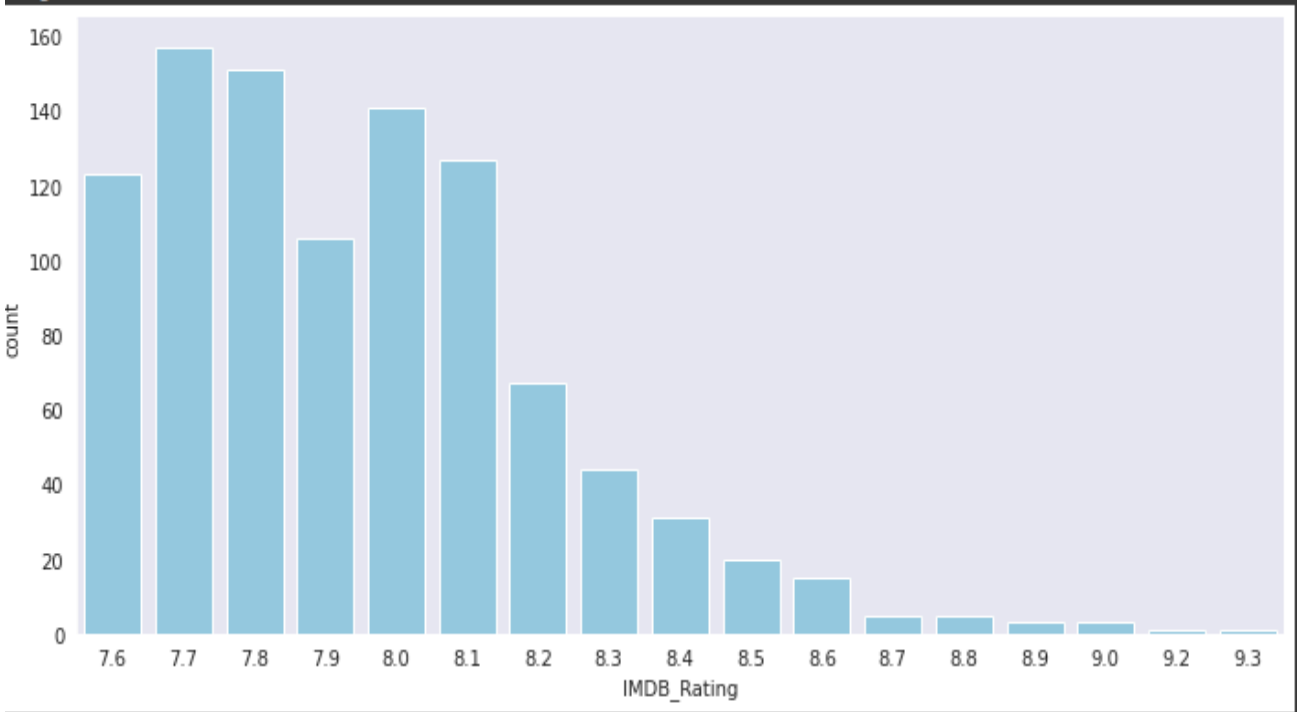
```
sns.heatmap(df.corr(),annot = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb5c8a7e490>
```



```
plt.figure(figsize=(20,20))
MovTrnd = sns.factorplot("IMDB_Rating", data=df, aspect=2, kind="count",
color='Skyblue')
```

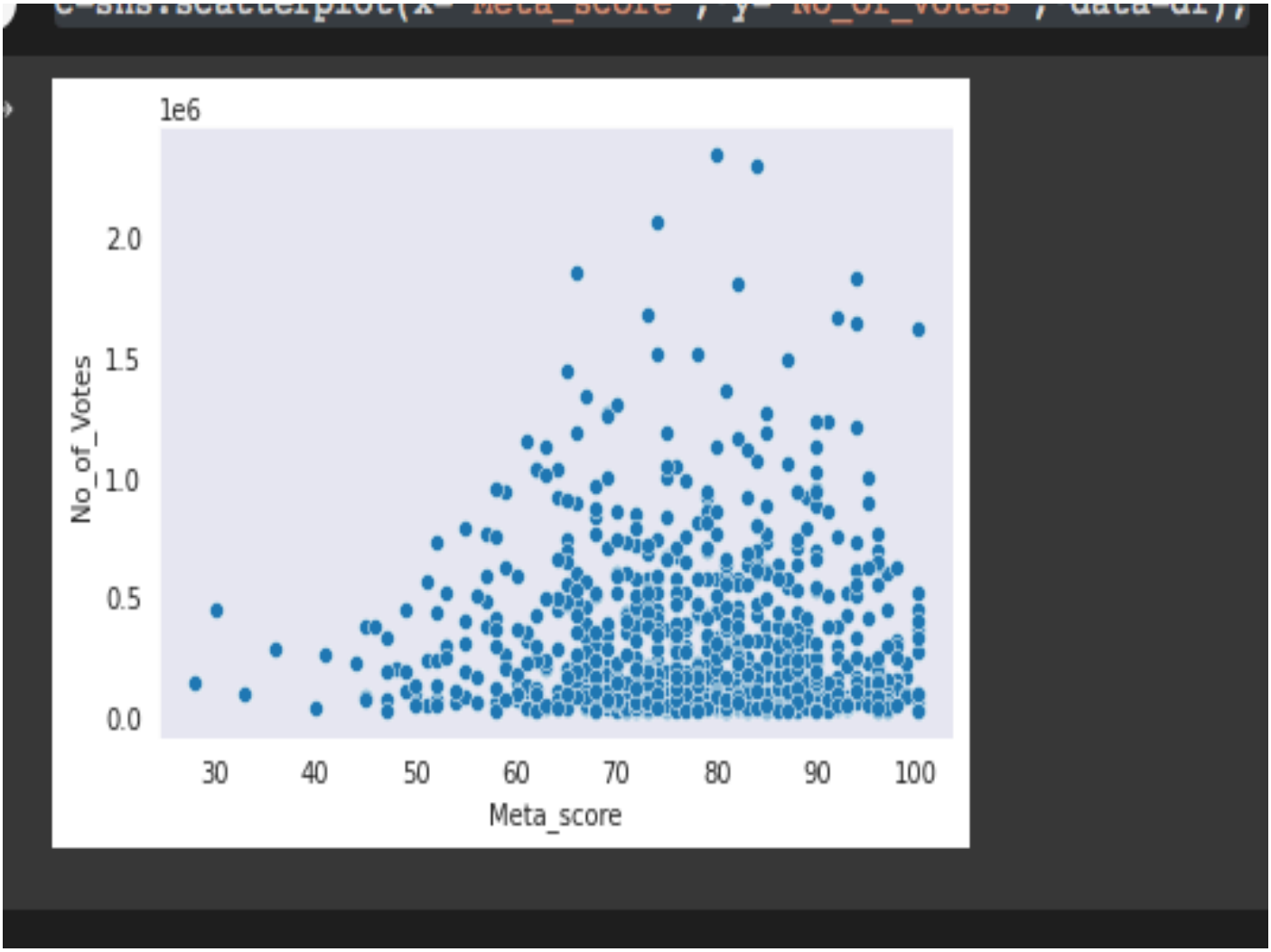
```
usr/local/lib/python3.7/dist-packages/seaborn/categorical.py:3717: UserWarning: The `factorplo
 warnings.warn(msg)
usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the follo
 FutureWarning
Figure size 1440x1440 with 0 Axes>
```




```
      df6 = (pd.DataFrame(list(zip(x, y)),
columns=['Director','Revenue_Sum'])).sort_values('Revenue_Sum',
ascending=False)
```

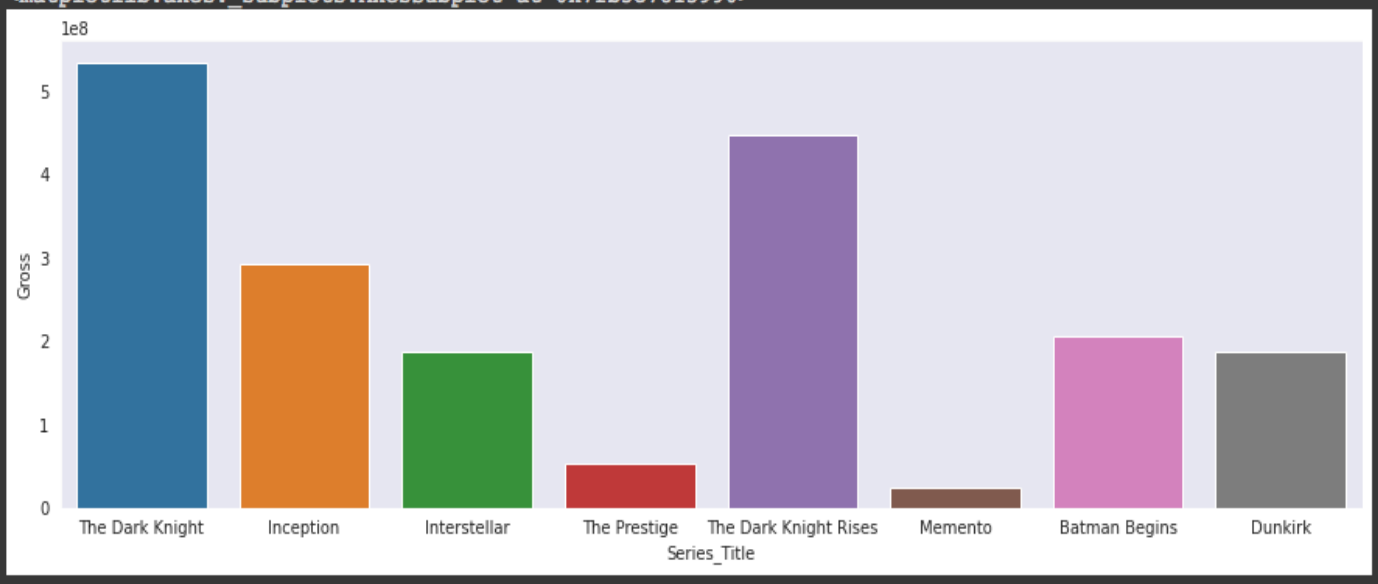
```
f7 = df6.tail(5)  #Lowest Revenue generated directors
df6 = df6.head(5)  #Highest Revenue generated directors
```

```
c=sns.scatterplot(x="Meta_score", y="No_of_Votes", data=df);
```




```
plt.figure(figsize=(15,5))
sns.barplot(x = 'Series_Title', y = 'Gross', data = df[df.Director ==
'Christopher Nolan'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb5c7815990>
```




```
plt.figure(figsize=(14,5))
sns.barplot(x = 'Director', y = 'Gross', data = df[df.Gross > 500000000],hue
= 'Certificate')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb5c7c83050>
```