In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
import warnings
warnings.filterwarnings("ignore")
```

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

Out[2]:

	Rank	Title	Genre	Description	Director	Actors	Year	R (M
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014	
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016	
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016	
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	2016	

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

Out[3]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)
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...	2015	111

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)
996	997	Hostel: Part II	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...	2007	94
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,...	2008	98
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...	2014	93
999	1000	Nine Lives	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	2016	87

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

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

In [5]:
```
data.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
```

In [6]:
```
data.describe()
```

Out[6]:

	Rank	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
count	1000.000000	1000.000000	1000.000000	1000.000000	1.000000e+03	872.000000	936.000000
mean	500.500000	2012.783000	113.172000	6.723200	1.698083e+05	82.956376	58.985043

	Rank	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
std	288.819436	3.205962	18.810908	0.945429	1.887626e+05	103.253540	17.194757
min	1.000000	2006.000000	66.000000	1.900000	6.100000e+01	0.000000	11.000000
25%	250.750000	2010.000000	100.000000	6.200000	3.630900e+04	13.270000	47.000000
50%	500.500000	2014.000000	111.000000	6.800000	1.107990e+05	47.985000	59.500000
75%	750.250000	2016.000000	123.000000	7.400000	2.399098e+05	113.715000	72.000000
max	1000.000000	2016.000000	191.000000	9.000000	1.791916e+06	936.630000	100.000000

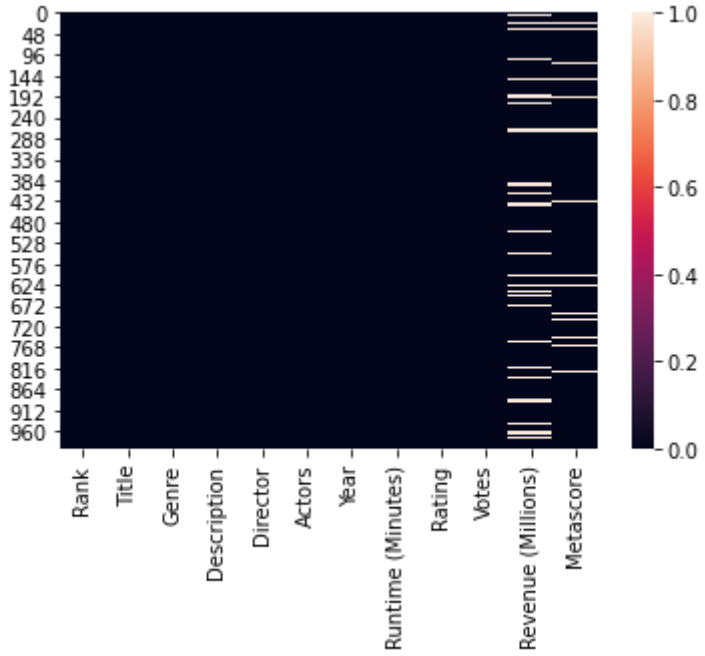
In [7]:
```
data.isnull()
```

Out[7]:

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

1000 rows × 12 columns

In [8]:
```
sns.heatmap(data.isnull())
plt.show()
```



In [9]:
```
data.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 = data.copy()
df.head()
```

Out[10]:

	Rank	Title	Genre	Description	Director	Actors	Year	R (M
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014	
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016	

	Rank	Title	Genre	Description	Director	Actors	Year	R (M
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016	
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	2016	

In [11]:
```
df['Revenue (Millions)'].fillna(df['Revenue (Millions)'].mean(), inplace = True)
df['Metascore'].fillna(df['Metascore'].mean(), inplace = True)
df.isnull().sum()
```

Out[11]:
```
Rank                  0
Title                 0
Genre                 0
Description           0
Director              0
Actors                0
Year                  0
Runtime (Minutes)     0
Rating                0
Votes                 0
Revenue (Millions)    0
Metascore             0
dtype: int64
```

In [12]:
```
dup_data=data.duplicated().any()
```

In [13]:
```
print("Are there any duplicate values?",dup_data)
```

```
Are there any duplicate values? False
```

In [14]:
```
df.columns
```

Out[14]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')
```

Movies runtime more than 3hours

In [15]:
```
data_more3hr=data[data['Runtime (Minutes)']>=180]['Title']
pd.DataFrame(data_more3hr)
```

Out[15]:

	Title
82	The Wolf of Wall Street

	Title
88	The Hateful Eight
311	La vie d'Adèle
828	Grindhouse
965	Inland Empire

In [16]:
```
data.columns
```

Out[16]:
```
Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')
```

In [17]:
```
#left
```

In [18]:
```
data.nlargest(10,'Runtime (Minutes)')[['Title','Runtime (Minutes)']]
```

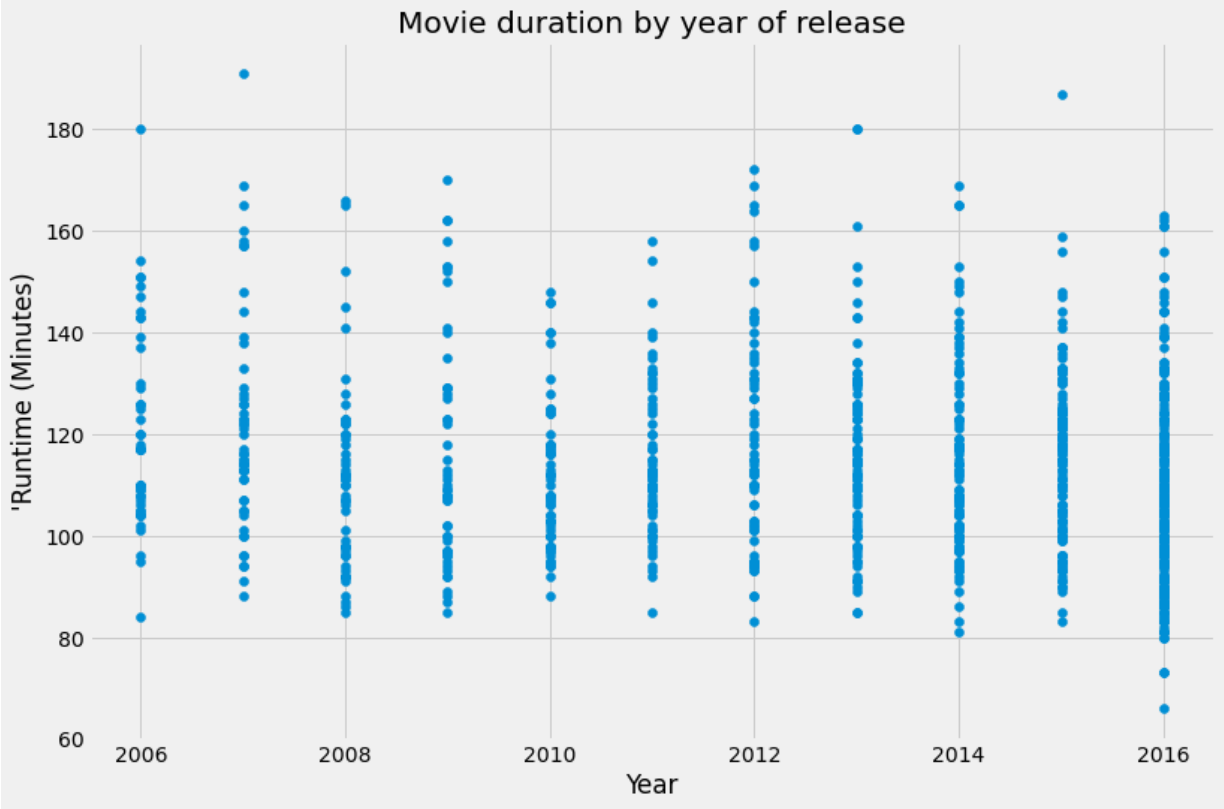
Out[18]:

	Title	Runtime (Minutes)
828	Grindhouse	191
88	The Hateful Eight	187
82	The Wolf of Wall Street	180
311	La vie d'Adèle	180
965	Inland Empire	180
267	Cloud Atlas	172
430	3 Idiots	170
36	Interstellar	169
75	Pirates of the Caribbean: At World's End	169
271	The Hobbit: An Unexpected Journey	169

In [19]:
```
plt.style.use('fivethirtyeight')
fig=plt.figure(figsize=(12,8))

#Create a scatter plot of duration versus release year

plt.scatter(

x=df['Year'],
y=df['Runtime (Minutes)']
)

#Create a title and axis labels
```

```
plt.title("Movie duration by year of release")
plt.xlabel("Year")
plt.ylabel("'Runtime (Minutes)")

#Show the plot
plt.plot()
```

Out[19]: []

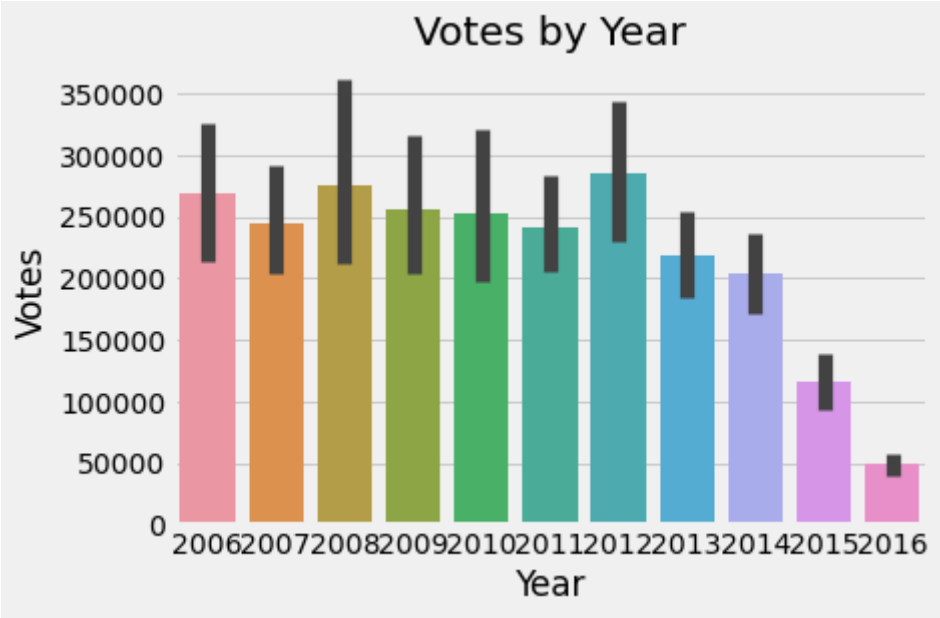


Mean of votes by years.

In [20]:
```
data.groupby('Year')["Votes"].mean().sort_values(ascending=False)
```

Out[20]:
```
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 [21]:
```
sns.barplot("Year",y="Votes",data=data)
plt.title("Votes by Year")
plt.show()
```



In [22]:
```
df.drop(['Genre'],axis=1,inplace=True )
df.drop(['Description'],axis=1,inplace=True )
df.drop(['Director'],axis=1,inplace=True )
df.drop(['Actors'],axis=1,inplace=True )
df.head()
```

Out[22]:

	Rank	Title	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
0	1	Guardians of the Galaxy	2014	121	8.1	757074	333.13	76.0
1	2	Prometheus	2012	124	7.0	485820	126.46	65.0
2	3	Split	2016	117	7.3	157606	138.12	62.0
3	4	Sing	2016	108	7.2	60545	270.32	59.0
4	5	Suicide Squad	2016	123	6.2	393727	325.02	40.0

In [23]:
```
df.head()
```

Out[23]:

	Rank	Title	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
0	1	Guardians of the Galaxy	2014	121	8.1	757074	333.13	76.0
1	2	Prometheus	2012	124	7.0	485820	126.46	65.0
2	3	Split	2016	117	7.3	157606	138.12	62.0
3	4	Sing	2016	108	7.2	60545	270.32	59.0
4	5	Suicide Squad	2016	123	6.2	393727	325.02	40.0

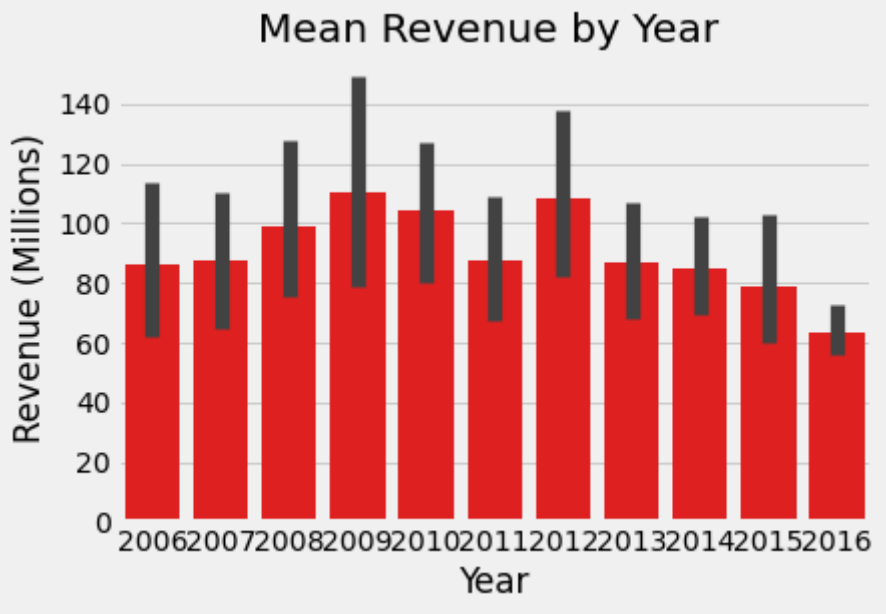
Mean Revenue by years

In [24]:
```
df.groupby("Year")["Revenue (Millions)"].mean()
```

Out[24]:
```
Year
2006     86.144835
2007     87.510481
2008     98.772623
2009    110.276186
2010    103.975319
2011     87.538355
2012    107.973281
2013     86.984496
2014     84.992097
2015     78.862278
2016     63.446588
Name: Revenue (Millions), dtype: float64
```

In [25]:
```
x=df["Year"]
y=df["Revenue (Millions)"]
```

In [26]:
```
sns.barplot(x,y ,color="red")
plt.title("Mean Revenue by Year ")
plt.show()
```



Top 10 Movies by Revenue

In [27]:
```
b=df.groupby("Title")["Revenue
(Millions)"].max().sort_values(ascending=False).head(10)
pd.DataFrame(b)
```

Out[27]:

Title	Revenue (Millions)
Star Wars: Episode VII - The Force Awakens	936.63
Avatar	760.51

Title	Revenue (Millions)
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

Top 10 Movies by Votes.

In [28]:
```
b=df.groupby("Title")
["Votes"].max().sort_values(ascending=False).head(10)
pd.DataFrame(b)
```

Out[28]:

Title	Votes
The Dark Knight	1791916
Inception	1583625
The Dark Knight Rises	1222645
Interstellar	1047747
The Avengers	1045588
Django Unchained	1039115
Inglourious Basterds	959065
The Departed	937414
Avatar	935408
The Prestige	913152

Top 10 by Rating.

In [29]:
```
df_rating=df.groupby("Title")
["Rating"].max().sort_values(ascending=False).head(10)
pd.DataFrame(df_rating)
```

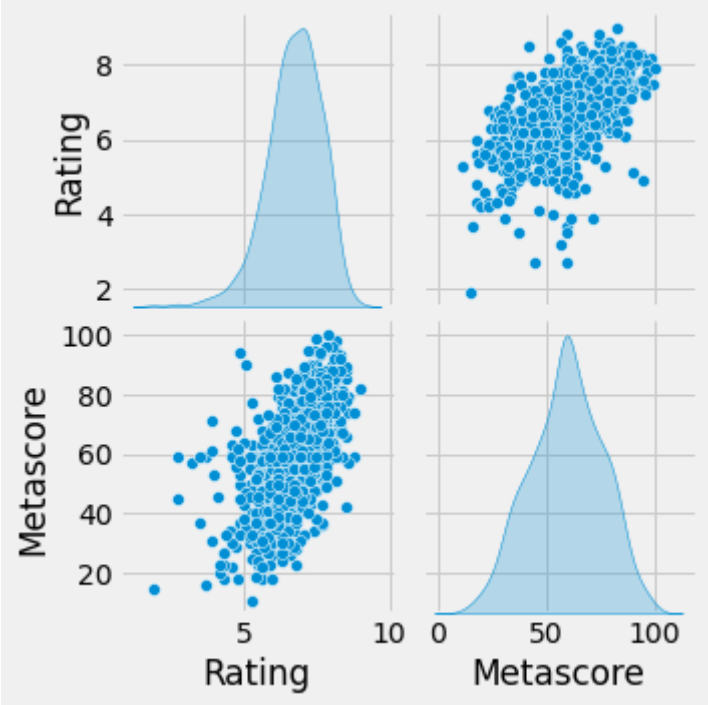
Out[29]:

Title	Rating

Title	Rating
The Dark Knight	9.0
Dangal	8.8
Inception	8.8
Interstellar	8.6
The Intouchables	8.6
Kimi no na wa	8.6
Taare Zameen Par	8.5
The Lives of Others	8.5
The Prestige	8.5
The Departed	8.5

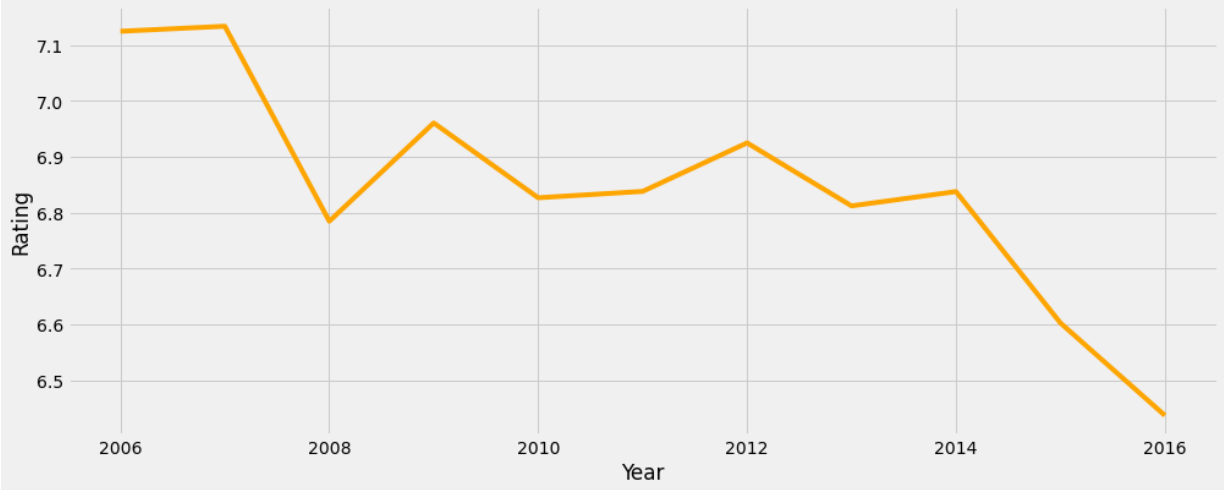
Pairplot for Rating & Metascore.

In [30]:
```
sns.pairplot(df[["Rating","Metascore"]],diag_kind="kde")
plt.show()
```



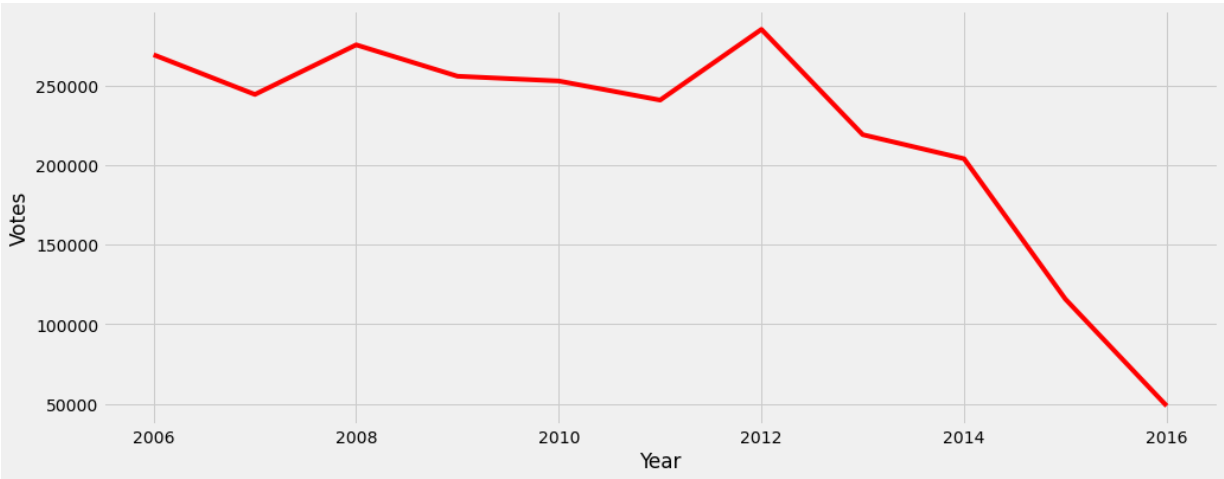
Year by rating.

In [31]:
```
plt.figure(figsize=(15,6))
sns.lineplot(df['Year'],df['Rating'], err_style=None,color="orange")
plt.show()
```



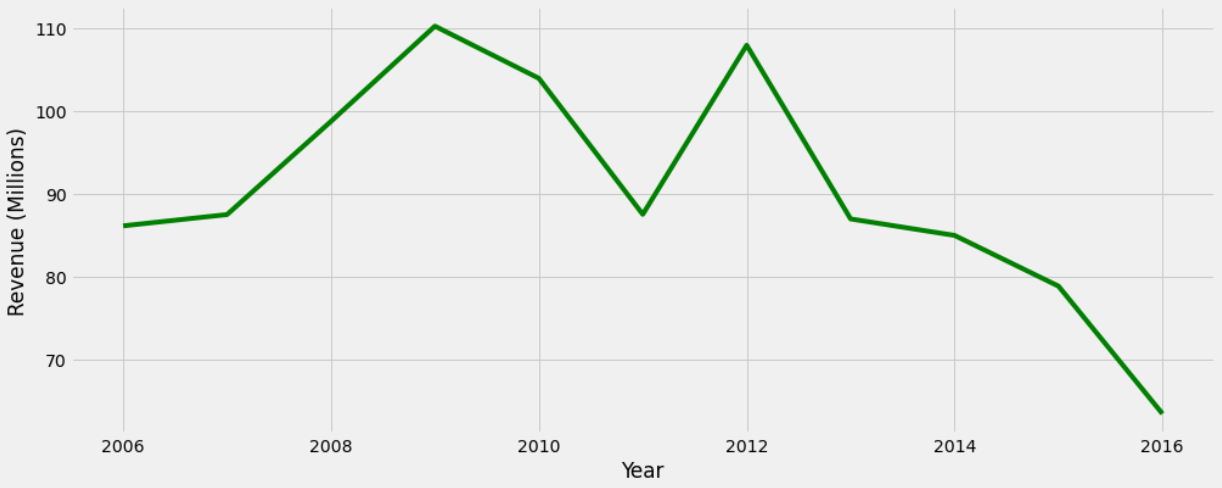
Year by Votes.

In [32]:
```
plt.figure(figsize=(15,6))
sns.lineplot(df['Year'],df['Votes'], err_style=None,color="red")
plt.show()
```

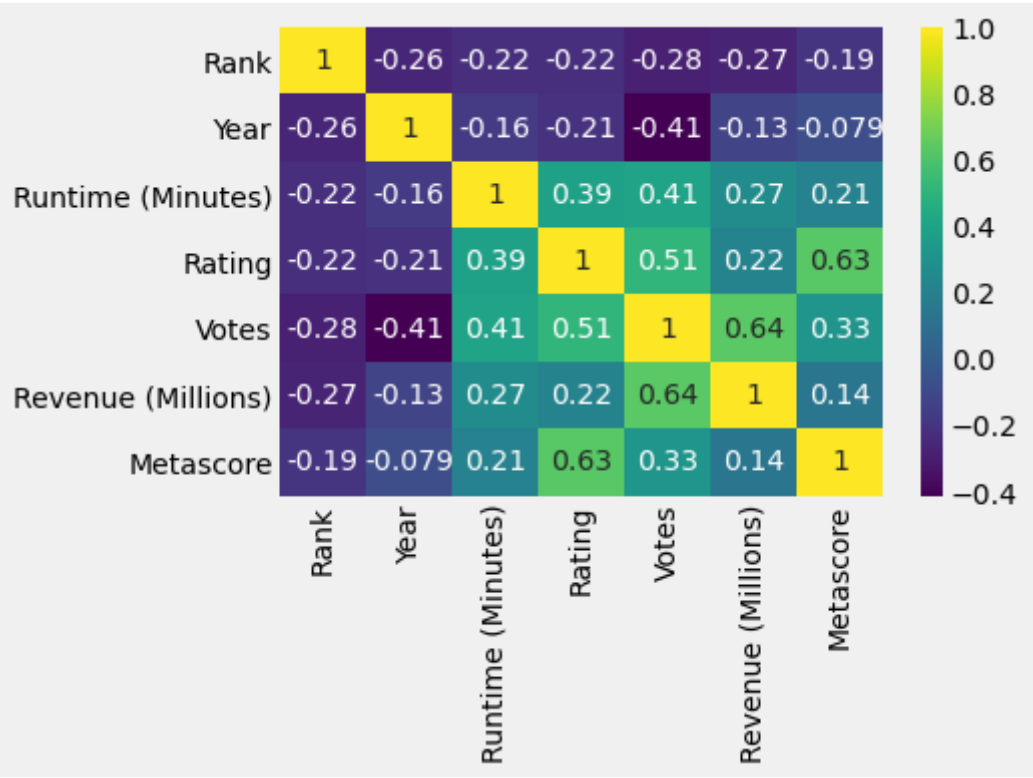


Year by Revenue.

In [33]:
```
plt.figure(figsize=(15,6))
sns.lineplot(df['Year'],df['Revenue (Millions)'],
err_style=None,color="green")
plt.show()
```



In [34]:
```
sns.heatmap(data.corr(), annot = True, cmap = 'viridis')
plt.show()
```



In [35]:
```
from sklearn import linear_model
```

In [36]:
```
from sklearn.linear_model import LinearRegression
```

In [38]:
```
lr=LinearRegression()
```

In [39]:
```
df.columns
```

Out[39]:
```
Index(['Rank', 'Title', 'Year', 'Runtime (Minutes)', 'Rating', 'Votes',
       'Revenue (Millions)', 'Metascore'],
      dtype='object')
```

In [99]:
```
x=df[["Votes"]]
```

```
y=df[["Revenue (Millions)"]]
```

In [100…
```
from sklearn.model_selection import train_test_split
```

In [101…
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_st
```

In [102…
```
x_train.head()
```

Out[102…

	Votes
105	115546
68	291
479	41642
399	113686
434	223

In [103…
```
lr.fit(x_train,y_train)
```

Out[103… LinearRegression()

In [104…
```
y_pred=lr.predict(x_test)
```

In [105…
```
lr.score(x_train,y_train)
```

Out[105… 0.33769779991188953

In [106…
```
print("Coefficient: \n",lr.coef_)
```

```
Coefficient:
 [[0.00028465]]
```

In [107…
```
print("Intercept :\n",lr.intercept_)
```

```
Intercept :
 [33.61382523]
```

In [108…
```
lr.score(y_test, y_pred)
```

Out[108… -0.7440714625299043

In [109…
```
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
```

In [110…

```
print("R2Score : " ,r2_score(y_test, y_pred))
print("mean_absolute_error : ",mean_absolute_error(y_test, y_pred))
print("mean_squared_error : " ,mean_squared_error(y_test, y_pred))
print("Root mean_squared_error : ",np.sqrt(mean_squared_error(y_test,
y_pred)))
```

```
R2Score :  0.4149415654872476
mean_absolute_error :  52.740548367277746
mean_squared_error :  6819.744965698331
Root mean_squared_error :  82.58174717029382
```

In [111…

```
a={"Actual value":x_train,"predicted value":y_pred}
a
```

Out[111…

```
{'Actual value':       Votes
 105  115546
 68      291
 479   41642
 399  113686
 434     223
 ..      ...
 835   38804
 192  268282
 629   74886
 559  115355
 684  245144

 [700 rows x 1 columns],
 'predicted value': array([[ 73.72104506],
        [ 92.47750379],
        [ 46.11822594],
        [ 36.22890705],
        [133.81725919],
        [ 88.6910862 ],
        [ 33.76269731],
        [ 34.5545943 ],
        [ 49.89781192],
        [ 93.0009756 ],
        [112.85447638],
        [ 59.33823737],
        [ 42.54330348],
        [ 61.18277097],
        [183.23908905],
        [ 47.61577089],
        [ 37.13238694],
        [ 74.33845144],
        [ 36.06523316],
        [306.61191452],
        [ 69.48829588],
        [ 60.31373376],
        [101.13884151],
        [ 62.97549823],
        [ 83.11991157],
        [ 53.29938237],
        [146.05778911],
        [ 35.66444561],
        [ 59.58218263],
        [171.90260831],
        [ 35.28045243],
        [129.18713827],
        [ 90.53618911],
        [ 47.9149383 ],
        [ 91.22134225],
        [ 34.79654701],
```

```
[ 88.32075623],
[ 88.92962311],
[ 90.36938406],
[118.6249062 ],
[175.61985588],
[ 54.43656011],
[121.6840424 ],
[132.54885769],
[ 48.11561672],
[ 33.90729964],
[ 33.77436797],
[ 43.34430928],
[ 54.41976575],
[ 74.29034555],
[134.1821808 ],
[ 39.4531404 ],
[ 39.79500535],
[174.46417588],
[ 42.81087472],
[ 59.1996127 ],
[156.59582591],
[189.42738542],
[ 43.20312276],
[119.23747353],
[ 53.1562033 ],
[ 34.83895989],
[103.74026012],
[ 56.34457072],
[ 43.69528303],
[ 34.69521152],
[ 50.49700069],
[ 37.24539308],
[ 73.41675394],
[ 46.25941246],
[ 38.57784089],
[147.66805556],
[ 42.05142786],
[ 37.80046107],
[ 59.69433483],
[105.37956089],
[ 96.42674132],
[ 58.6308815 ],
[ 45.10145526],
[ 33.75102665],
[199.14306766],
[102.41635182],
[ 62.38968802],
[ 60.70455855],
[ 86.84456005],
[ 59.28586172],
[ 45.41143937],
[ 55.81512126],
[118.69948456],
[ 62.44917992],
[ 69.66648693],
[ 60.57219619],
[ 50.1010522 ],
[201.84866826],
[ 47.96076699],
[106.04194202],
[ 41.16303444],
[ 33.89961408],
[ 33.6952352 ],
[ 34.62831871],
[132.14664689],
[ 65.94980866],
[158.91942588],
[ 49.33875883],
[ 33.97333849],
```

```
[148.8641559 ],
[ 41.10581974],
[137.83509743],
[ 54.55468996],
[159.88012047],
[ 36.24598607],
[ 50.70963442],
[106.43105891],
[ 43.90563957],
[207.69538433],
[ 77.634132  ],
[ 49.20895832],
[ 46.66987812],
[177.08865116],
[ 47.07465077],
[ 95.86085663],
[ 87.93448585],
[ 48.75237933],
[ 60.1779556 ],
[168.05612952],
[ 78.44595451],
[ 44.55293423],
[ 65.14538706],
[113.22879146],
[ 43.16668753],
[194.6464477 ],
[ 76.74374603],
[ 57.54807196],
[ 60.04730113],
[ 77.56382339],
[ 74.80698575],
[ 71.352755  ],
[ 35.65989121],
[103.45959498],
[ 88.71613543],
[196.09617141],
[ 39.61681429],
[ 40.47389619],
[192.43870039],
[ 43.95004501],
[ 62.27867442],
[109.4753933 ],
[ 65.57663218],
[543.68315752],
[ 46.75470389],
[223.94549764],
[ 47.31204907],
[150.49320927],
[ 33.68669569],
[ 39.58180231],
[ 70.53523949],
[ 60.68947209],
[152.50170141],
[ 67.59423314],
[ 38.61911518],
[ 35.55200876],
[ 58.66304698],
[200.60930108],
[ 35.37154051],
[ 35.74158583],
[ 51.20862631],
[ 60.92886295],
[ 33.98159335],
[ 61.41817672],
[135.29459397],
[ 54.26036161],
[ 84.35073924],
[ 43.12626719],
[ 35.24060139],
```

```
[122.93792674],
[ 60.05071694],
[128.18431545],
[ 41.30108981],
[ 74.46597475],
[329.39816681],
[133.62882072],
[ 37.15345105],
[ 65.3688375 ],
[ 90.27288763],
[ 52.866714  ],
[ 77.1106602 ],
[ 61.14462784],
[ 85.43981108],
[ 57.15468532],
[ 53.68280626],
[ 97.77228304],
[ 52.81746951],
[ 56.75617498],
[ 49.56448648],
[ 40.99423684],
[ 34.29783977],
[ 87.80696254],
[ 54.55497461],
[ 42.40240161],
[ 48.08914425],
[105.48886659],
[ 78.97682722],
[134.03871708],
[ 64.57580191],
[117.3038444 ],
[ 52.51460164],
[ 76.2484546 ],
[ 72.69601952],
[179.85118181],
[ 47.9189234 ],
[214.9656363 ],
[ 70.81818183],
[117.22471163],
[ 86.86533952],
[ 83.30037982],
[ 46.54235481],
[116.57713232],
[ 34.17544017],
[ 67.96826356],
[ 87.88894181],
[ 48.56849527],
[ 66.66684263],
[ 33.6935273 ],
[ 55.00159085],
[ 98.1329349 ],
[197.85986434],
[ 34.07780513],
[ 34.13473518],
[ 62.66551411],
[171.69054387],
[ 41.14595542],
[ 88.6876704 ],
[ 60.39058933],
[ 70.40543898],
[ 62.69853353],
[ 75.66150579],
[128.08981156],
[ 39.90658824],
[ 76.01048699],
[ 62.58581204],
[ 74.42953952],
[ 49.49190067],
[167.91608159],
```

```
       [ 58.96449159],
       [ 62.84655166],
       [ 35.95621211],
       [ 35.17826299],
       [ 78.47641208],
       [ 33.6311889 ],
       [ 84.16827843],
       [131.01630075],
       [ 72.41820087],
       [163.17779358],
       [ 87.94074815],
       [175.60334617],
       [ 33.72142302],
       [ 89.97628207],
       [135.35807097],
       [ 52.91339664],
       [ 68.58396205],
       [ 71.21555358],
       [ 93.19852287],
       [ 50.24337732],
       [183.43208192],
       [ 59.54717065],
       [ 72.80560986],
       [ 74.82520336],
       [ 57.50765163],
       [ 82.94940607],
       [ 33.66050787],
       [ 36.88217937],
       [ 75.88552553],
       [221.94070595],
       [190.82558744],
       [ 63.01221811],
       [118.18142111],
       [ 39.56500795],
       [ 43.87347409],
       [ 76.7124345 ],
       [ 34.21927631],
       [ 57.56942073],
       [ 43.3625269 ],
       [ 43.10748027],
       [ 49.45404218],
       [ 88.90315064],
       [ 33.68242594],
       [ 58.89902204],
       [ 96.76575976],
       [147.62165757],
       [ 72.43983429],
       [ 64.95381744],
       [ 79.55523652],
       [ 33.70292076],
       [ 61.26503489],
       [ 54.63097623],
       [ 44.920987  ],
       [ 35.00320309],
       [ 39.70676377],
       [102.31672423],
       [ 99.67317739]])}
```

In [113…

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train_std = sc.fit_transform(x_train) # study & exam
x_test_std = sc.transform(x_test)
```

In [114…

```
from sklearn.ensemble import RandomForestRegressor
```

```
rf_tree = RandomForestRegressor(random_state=0)
rf_tree.fit(x_train_std,y_train)
rf_tree_y_pred = rf_tree.predict(x_train_std)
print("Accuracy: {}".format(rf_tree.score(x_train_std,y_train)))
# or
print("R squared:
{}".format(r2_score(y_true=y_train,y_pred=rf_tree_y_pred)))
```

```
Accuracy: 0.8620132614862872
R squared: 0.8620132614862872
```