In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [2]:
```
data = pd.read_csv("Downloads/IMDb Movies India.csv.zip",encoding='ISO-8859-1'
data
```

Out[2]:

	Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Acto
0		NaN	NaN	Drama	NaN	NaN	J.S. Randhawa	Manmauji	Bir
1	#Gadhvi (He thought he was Gandhi)	(2019)	109 min	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vi Ghamar
2	#Homecoming	(2021)	90 min	Drama, Musical	NaN	NaN	Soumyajit Majumdar	Sayani Gupta	Plat Borthal
3	#Yaaram	(2019)	110 min	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita I
4	...And Once Again	(2010)	105 min	Drama	NaN	NaN	Amol Palekar	Rajat Kapoor	Ritupa Sengu
...	...	...	...	...	...	...	...	...	
15504	Zulm Ko Jala Doonga	(1988)	NaN	Action	4.6	11	Mahendra Shah	Naseeruddin Shah	Sum Sai
15505	Zulmi	(1999)	129 min	Action, Drama	4.5	655	Kuku Kohli	Akshay Kumar	Twin Khar
15506	Zulmi Raj	(2005)	NaN	Action	NaN	NaN	Kiran Thej	Sangeeta Tiwari	N
15507	Zulmi Shikari	(1988)	NaN	Action	NaN	NaN	NaN	NaN	N
15508	Zulm-O-Sitam	(1998)	130 min	Action, Drama	6.2	20	K.C. Bokadia	Dharmendra	Ja Pra

15509 rows × 10 columns

In [3]:
```
data.shape
```

Out[3]:
```
(15509, 10)
```

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

Out[4]:
```
Index(['Name', 'Year', 'Duration', 'Genre', 'Rating', 'Votes', 'Director',
       'Actor 1', 'Actor 2', 'Actor 3'],
      dtype='object')
```

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15509 entries, 0 to 15508
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Name      15509 non-null  object 
 1   Year      14981 non-null  object 
 2   Duration  7240 non-null   object 
 3   Genre     13632 non-null  object 
 4   Rating    7919 non-null   float64
 5   Votes     7920 non-null   object 
 6   Director  14984 non-null  object 
 7   Actor 1   13892 non-null  object 
 8   Actor 2   13125 non-null  object 
 9   Actor 3   12365 non-null  object 
dtypes: float64(1), object(9)
memory usage: 1.2+ MB
```

In [6]:
```
data.isnull().sum()
```

Out[6]:
```
Name           0
Year         528
Duration    8269
Genre       1877
Rating      7590
Votes       7589
Director     525
Actor 1     1617
Actor 2     2384
Actor 3     3144
dtype: int64
```

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

Out[7]:

	Rating
count	7919.000000
mean	5.841621
std	1.381777
min	1.100000
25%	4.900000
50%	6.000000
75%	6.800000
max	10.000000

In [8]:
```
data.dropna(inplace=True)
data.isnull().sum()
```

Out[8]:
```
Name        0
Year        0
Duration    0
Genre       0
Rating      0
Votes       0
Director    0
Actor 1     0
Actor 2     0
Actor 3     0
dtype: int64
```

In [9]:
```
data.shape
```

Out[9]:
```
(5659, 10)
```

In [10]:
```
data.head()
```

Out[10]:

	Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor 2	Actor 3
1	#Gadhvi (He thought he was Gandhi)	(2019)	109 min	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vivek Ghamande	Arvind Jangid
3	#Yaaram	(2019)	110 min	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita Raj	Siddhant Kapoor
5	...Aur Pyaar Ho Gaya	(1997)	147 min	Comedy, Drama, Musical	4.7	827	Rahul Rawail	Bobby Deol	Aishwarya Rai Bachchan	Shammi Kapoor
6	...Yahaan	(2005)	142 min	Drama, Romance, War	7.4	1,086	Shoojit Sircar	Jimmy Sheirgill	Minissha Lamba	Yashpal Sharma
8	?: A Question Mark	(2012)	82 min	Horror, Mystery, Thriller	5.6	326	Allyson Patel	Yash Dave	Muntazir Ahmad	Kiran Bhatia

In [11]:
```
data['Votes'] = data['Votes'].str.replace(',','').astype('int')
```

In [12]:
```
data['Year'] = data['Year'].str.strip('()').astype(int)
```

In [13]:
```
data['Duration'] = data['Duration'].str.strip('min')
```

In [14]:
```
data.info()
```

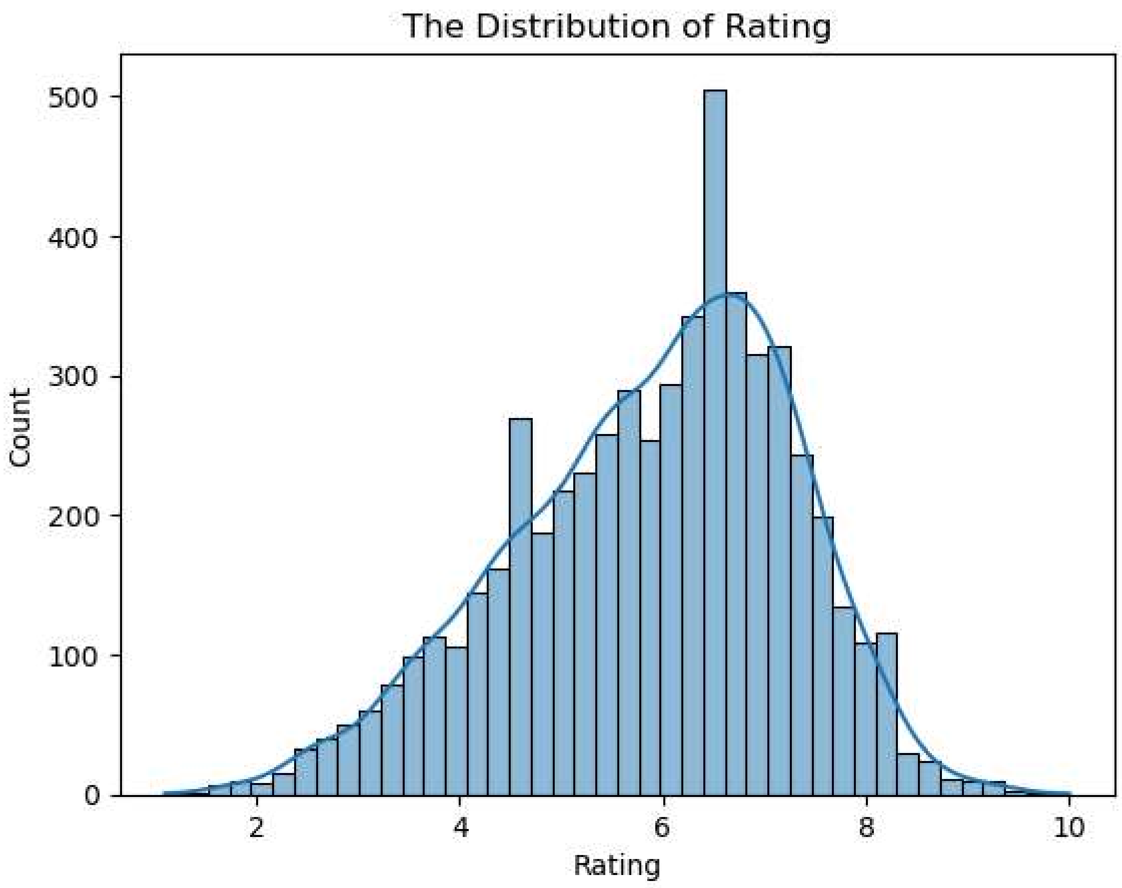
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5659 entries, 1 to 15508
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Name      5659 non-null   object 
 1   Year      5659 non-null   int32  
 2   Duration  5659 non-null   object 
 3   Genre     5659 non-null   object 
 4   Rating    5659 non-null   float64
 5   Votes     5659 non-null   int32  
 6   Director  5659 non-null   object 
 7   Actor 1   5659 non-null   object 
 8   Actor 2   5659 non-null   object 
 9   Actor 3   5659 non-null   object 
dtypes: float64(1), int32(2), object(7)
memory usage: 442.1+ KB
```

In [15]:
```
data.head()
```

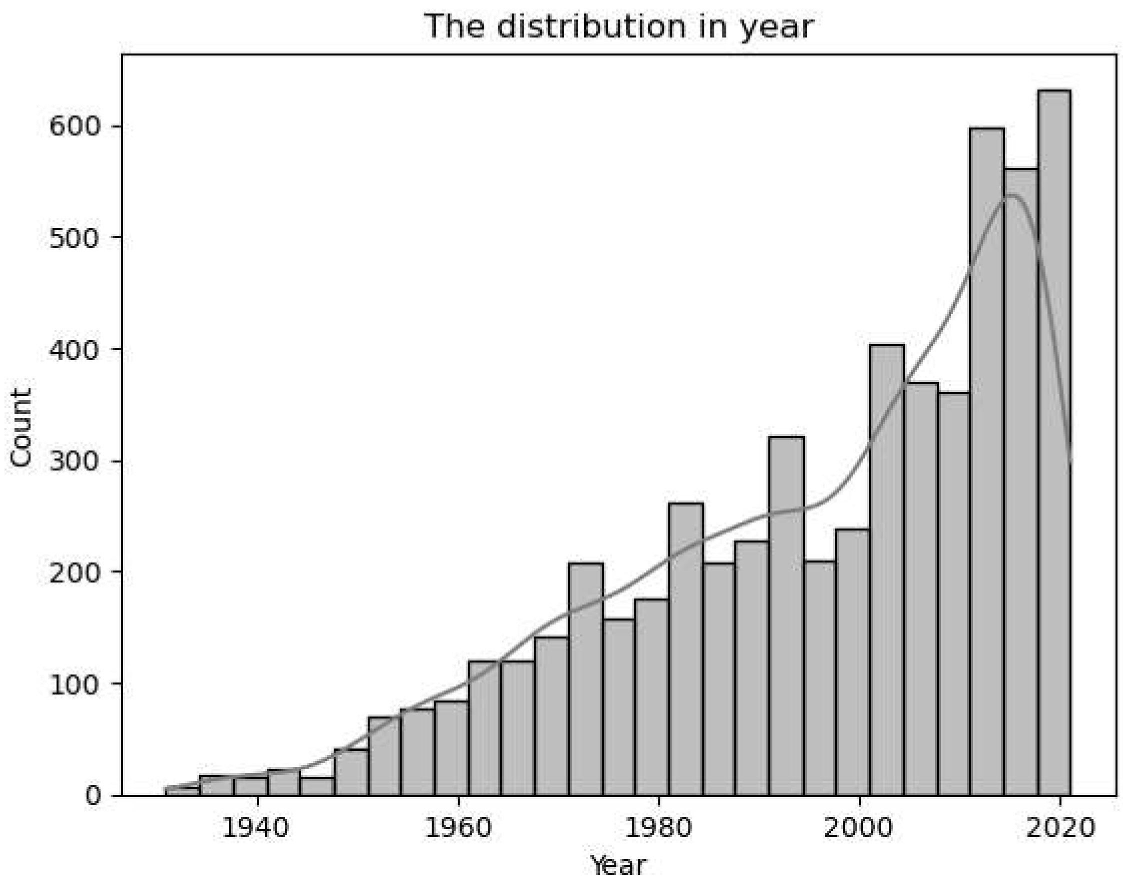
Out[15]:

	Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor 2	Actor 3
1	#Gadhvi (He thought he was Gandhi)	2019	109	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vivek Ghamande	Arvind Jangid
3	#Yaaram	2019	110	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita Raj	Siddhant Kapoor
5	...Aur Pyaar Ho Gaya	1997	147	Comedy, Drama, Musical	4.7	827	Rahul Rawail	Bobby Deol	Aishwarya Rai Bachchan	Shammi Kapoor
6	...Yahaan	2005	142	Drama, Romance, War	7.4	1086	Shoojit Sircar	Jimmy Sheirgill	Minissha Lamba	Yashpal Sharma
8	?: A Question Mark	2012	82	Horror, Mystery, Thriller	5.6	326	Allyson Patel	Yash Dave	Muntazir Ahmad	Kiran Bhatia

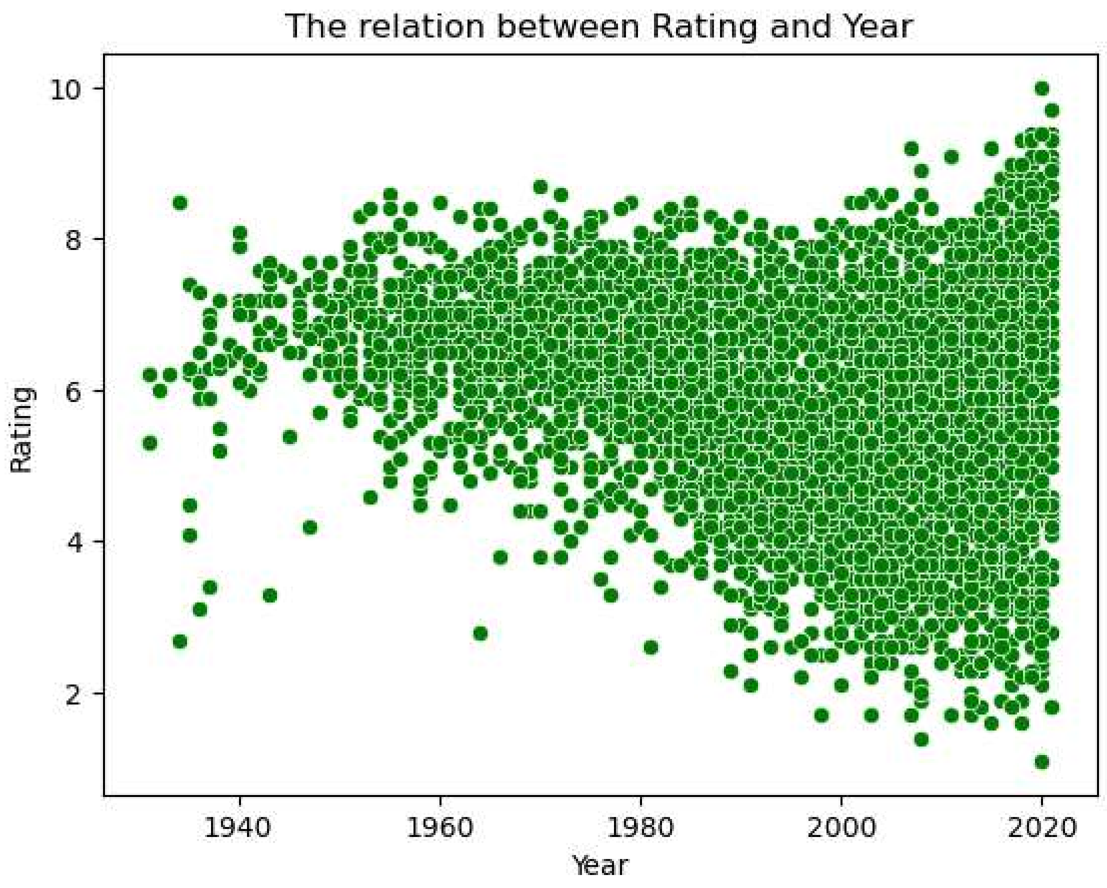
In [16]:
```python
sns.histplot(data = data, x='Rating', kde = True)
plt.title("The Distribution of Rating")
plt.show()
```



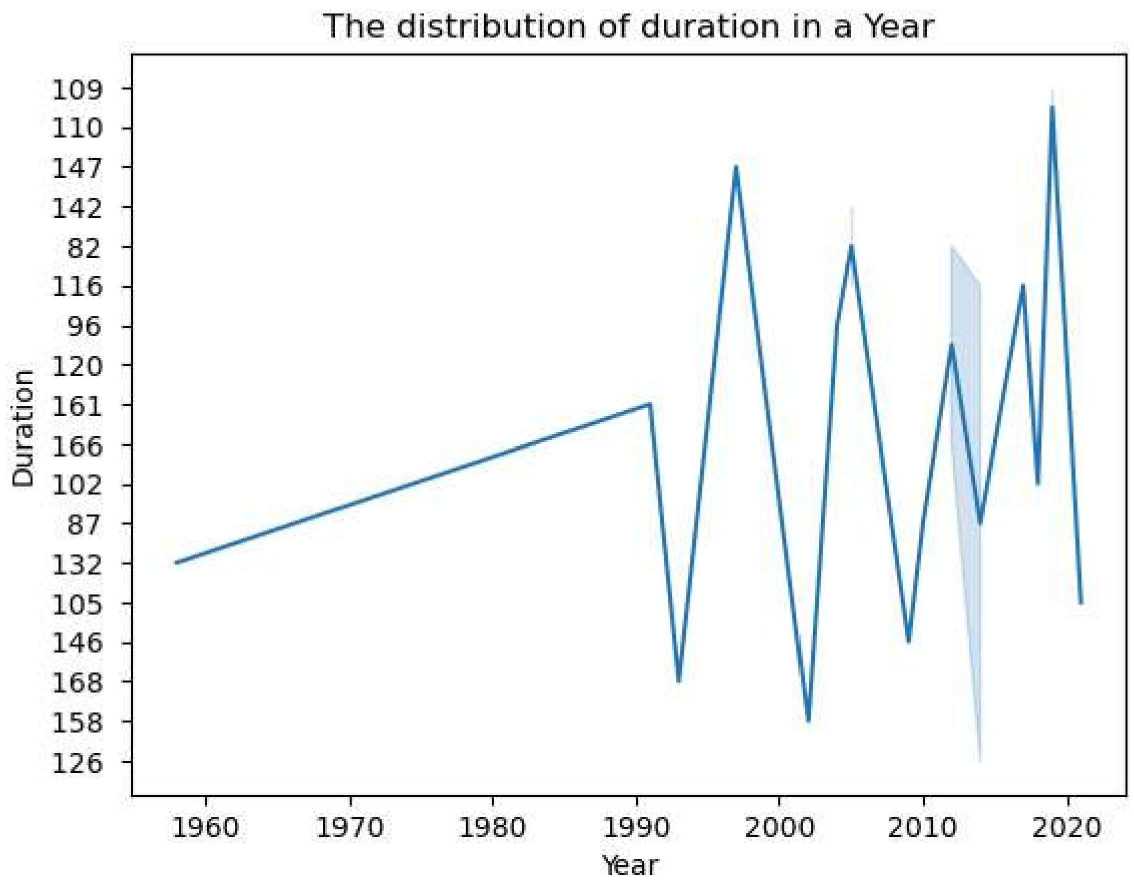
In [17]:
```
sns.histplot(data=data, x='Year',color='grey', kde=True)
plt.title("The distribution in year")
plt.show()
```



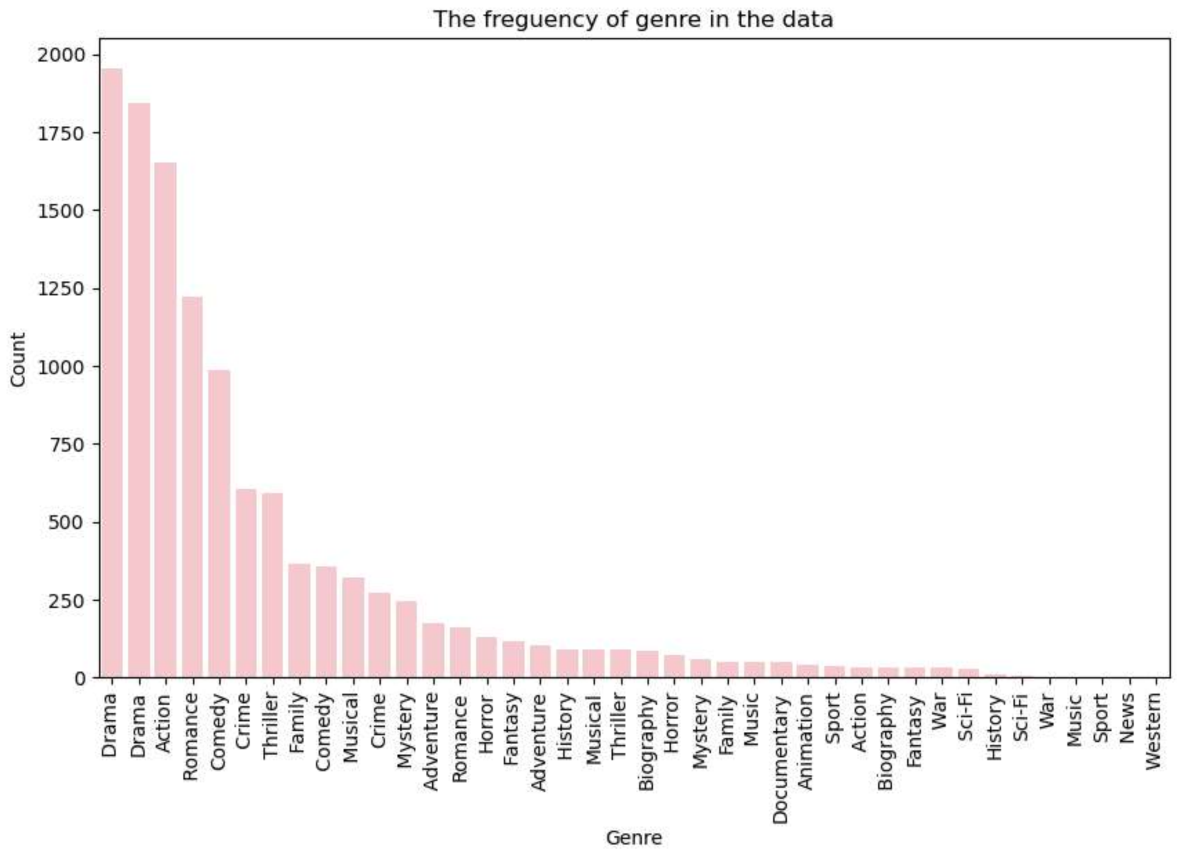
In [18]:
```python
sns.scatterplot(data=data, x='Year', y='Rating', color='green')
plt.title("The relation between Rating and Year")
plt.show()
```



In [19]:
```
sns.lineplot(data=data.head(20), x='Year', y='Duration')
plt.title("The distribution of duration in a Year")
plt.show()
```



In [20]:
```
from sklearn.preprocessing import LabelEncoder
movies_genre = data['Genre'].str.split(',',expand=True).stack().value_counts()
labels = movies_genre.keys()
count = movies_genre.values
plt.figure(figsize=(10,6))
sns.barplot(x=labels, y=count, color = ("pink"))
plt.xticks(rotation = 90)
plt.title("The freguency of genre in the data")
plt.xlabel("Genre")
plt.ylabel("Count")
plt.show()
```



In [21]:
```
encoder = LabelEncoder()
data['Actor 1'] = encoder.fit_transform(data['Actor 1'])
data['Actor 2'] = encoder.fit_transform(data['Actor 2'])
data['Actor 3'] = encoder.fit_transform(data['Actor 3'])
data['Director'] = encoder.fit_transform(data['Director'])
data['Genre'] = encoder.fit_transform(data['Genre'])
data.head()
```

Out[21]:

	Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor 2	Actor 3
1	#Gadhvi (He thought he was Gandhi)	2019	109	229	7.0	8	629	1352	2272	319
3	#Yaaram	2019	110	184	4.4	35	1335	1198	719	2148
5	...Aur Pyaar Ho Gaya	1997	147	157	4.7	827	1530	378	75	2045
6	...Yahaan	2005	142	289	7.4	1086	2044	692	1112	2524
8	?: A Question Mark	2012	82	320	5.6	326	135	1934	1175	1013

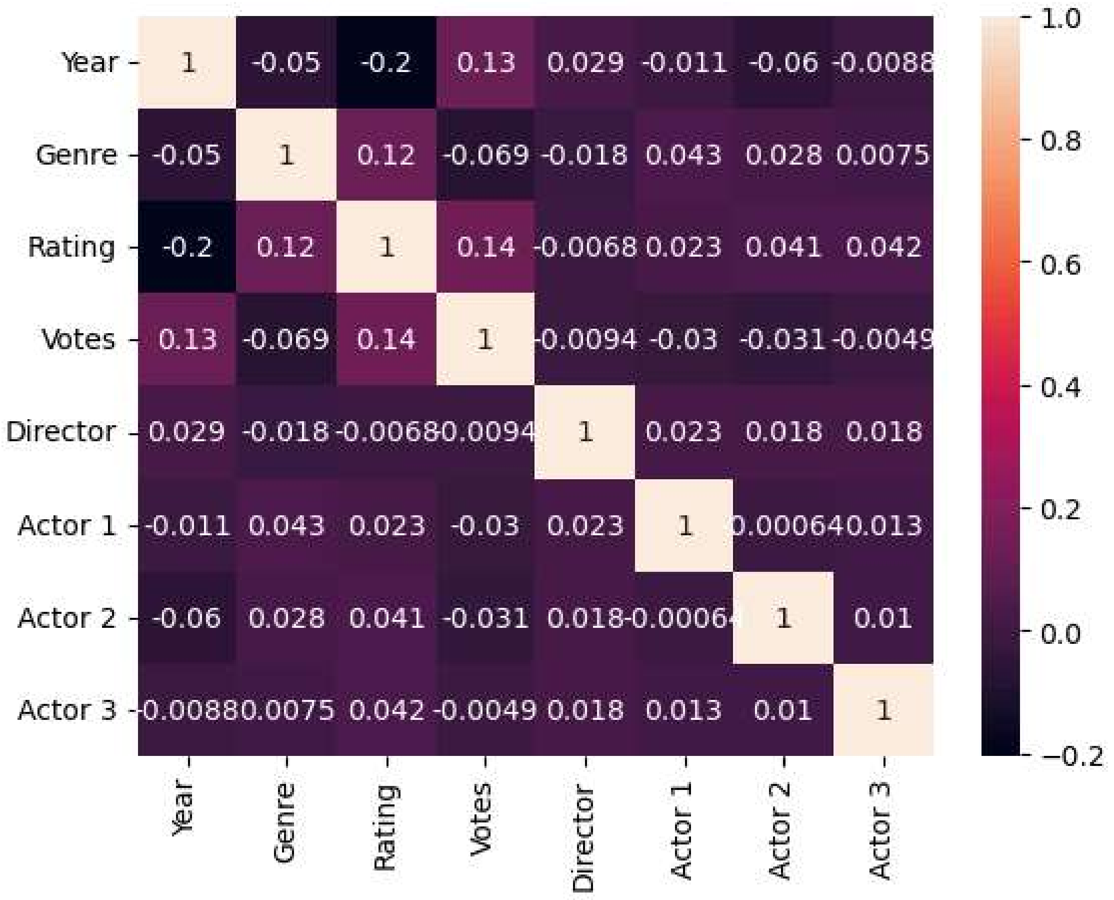
In [22]:
```
data1 = data.drop('Name',axis=1)
data1.head()
```

Out[22]:

	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor 2	Actor 3
1	2019	109	229	7.0	8	629	1352	2272	319
3	2019	110	184	4.4	35	1335	1198	719	2148
5	1997	147	157	4.7	827	1530	378	75	2045
6	2005	142	289	7.4	1086	2044	692	1112	2524
8	2012	82	320	5.6	326	135	1934	1175	1013

In [23]:
```
sns.heatmap(data1.corr(), annot=True)
plt.show()
```

```
C:\Users\Mano\AppData\Local\Temp\ipykernel_22396\2170505126.py:1: FutureWarni
ng: The default value of numeric_only in DataFrame.corr is deprecated. In a f
uture version, it will default to False. Select only valid columns or specify
the value of numeric_only to silence this warning.
  sns.heatmap(data1.corr(), annot=True)
```



	Year	Genre	Rating	Votes	Director	Actor 1	Actor 2	Actor 3
Year	1	-0.05	-0.2	0.13	0.029	-0.011	-0.06	-0.0088
Genre	-0.05	1	0.12	-0.069	-0.018	0.043	0.028	0.0075
Rating	-0.2	0.12	1	0.14	-0.0068	0.023	0.041	0.042
Votes	0.13	-0.069	0.14	1	-0.0094	-0.03	-0.031	-0.0049
Director	0.029	-0.018	-0.0068	0.0094	1	0.023	0.018	0.018
Actor 1	-0.011	0.043	0.023	-0.03	0.023	1	0.00064	0.013
Actor 2	-0.06	0.028	0.041	-0.031	0.018	-0.00064	1	0.01
Actor 3	-0.0088	0.0075	0.042	-0.0049	0.018	0.013	0.01	1

In [24]:
```
x = data1.drop('Rating', axis=1)
Y = data1['Rating']
```

In [25]:
```
x_train, x_test, Y_train, Y_test = train_test_split(x,Y, test_size = 0.3, ranc
```

In [26]:
```
model = LinearRegression()
model.fit(x_train,Y_train)
```

Out[26]:
```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [27]:
```
pred = model.predict(x_test)
pred
```

Out[27]:
```
array([5.63683425, 5.84694164, 5.93283536, ..., 6.02786817, 5.89959597,
       5.36480764])
```

In [28]:
```
from sklearn.metrics import confusion_matrix , accuracy_score , mean_absolute_
print("The Mean Absolute Error is :", mean_absolute_error(Y_test,pred))
print("The mean Squared Error is : ", mean_squared_error(Y_test,pred))
```

```
The Mean Absolute Error is : 1.0355603579570467
The mean Squared Error is :  1.6737798290839636
```

In [29]:
```
print("The R2 Score is : ", r2_score(Y_test,pred))
```

```
The R2 Score is :  0.08633120473998968
```

In [ ]: