In [1]:

```
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
movie=pd.read_csv("C:\\Users\\USER\\Documents\\Movie dataset.csv",e
movie
```

Out[2]:

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

15509 rows × 10 columns

In [3]:
```
1 movie.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 [4]:
```
1 movie.isnull().sum()
```

Out[4]:
```
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 [5]:
```
1 movie.shape
```

Out[5]:
```
(15509, 10)
```

In [6]:
```
1 movie.describe()
```

Out[6]:

	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 [7]:
```
1 movie.drop_duplicates()
```

Out[7]:

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

15503 rows × 10 columns

In [8]:
```
1 movie.columns
```

Out[8]:
```
Index(['Name', 'Year', 'Duration', 'Genre', 'Rating', 'Votes', 'Director',
       'Actor 1', 'Actor 2', 'Actor 3'],
      dtype='object')
```

In [9]:
```
1 movie['Name'].unique()
```

Out[9]:
```
array([' ', '#Gadhvi (He thought he was Gandhi)', '#Homecoming', ...,
       'Zulmi Raj', 'Zulmi Shikari', 'Zulm-O-Sitam'], dtype=object)
```

In [10]:
```
1 movie['Name'].nunique()
```

Out[10]:
```
13838
```

In [11]:
```
1 movie['Name'].mode()
```

Out[11]:
```
0    Anjaam
1     Mamta
Name: Name, dtype: object
```

In [12]:
```
movie['Name'].replace(to_replace=[', ...,','Anjaam'], value='new_va
movie['Name'].replace(to_replace=[' ', 'Anjaam'], value='new_value'
movie['Name'].replace(to_replace=['Gadhvi (He thought he was Gandhi
movie['Name'].replace(to_replace=['Yaaram', ..., 'Zulmi Raj','Yaara
```

In [13]:
```
movie['Name']=movie['Name'].str.strip('#')
```

In [14]:
```
movie['Name'].unique()
```

Out[14]:
```
array(['new_value', 'Gadhvi (He thought he was Gandhi)', 'Homecoming',
       ..., 'Zulmi', 'Zulmi Shikari', 'Zulm-O-Sitam'], dtype=object)
```

In [15]:
```
movie['Year']=movie['Year'].str.strip('(')
```

In [16]:
```
movie['Year']=movie['Year'].str.strip(')')
```

In [17]:
```
movie['Year'].unique()
```

Out[17]:
```
array([nan, '2019', '2021', '2010', '1997', '2005', '2008', '2012',
       '2014', '2004', '2016', '1991', '1990', '2018', '1987', '1948',
       '1958', '2017', '2020', '2009', '2002', '1993', '1946', '1994',
       '2007', '2013', '2003', '1998', '1979', '1951', '1956', '1974',
       '2015', '2006', '1981', '1985', '2011', '2001', '1967', '1988',
       '1995', '1959', '1996', '1970', '1976', '2000', '1999', '1973',
       '1968', '1943', '1953', '1986', '1983', '1989', '1982', '1977',
       '1957', '1950', '1992', '1969', '1975', '1947', '1972', '1971',
       '1935', '1978', '1960', '1944', '1963', '1940', '1984', '1934',
       '1955', '1936', '1980', '1966', '1949', '1962', '1964', '1952',
       '1933', '1942', '1939', '1954', '1945', '1961', '1965', '1938',
       '1941', '1931', '1937', '2022', '1932', '1923', '1915', '1928',
       '1922', '1917', '1913', '1930', '1926', '1914', '1924'],
      dtype=object)
```

In [18]:
```
movie['Year'].fillna('2019',inplace=True)
```

In [19]:
```
movie['Year']=movie['Year'].astype(int)
```

In [20]:
```
1 movie['Year'].unique()
```

Out[20]:
```
array([2019, 2021, 2010, 1997, 2005, 2008, 2012, 2014, 2004, 2016, 199
1,
       1990, 2018, 1987, 1948, 1958, 2017, 2020, 2009, 2002, 1993, 194
6,
       1994, 2007, 2013, 2003, 1998, 1979, 1951, 1956, 1974, 2015, 200
6,
       1981, 1985, 2011, 2001, 1967, 1988, 1995, 1959, 1996, 1970, 197
6,
       2000, 1999, 1973, 1968, 1943, 1953, 1986, 1983, 1989, 1982, 197
7,
       1957, 1950, 1992, 1969, 1975, 1947, 1972, 1971, 1935, 1978, 196
0,
       1944, 1963, 1940, 1984, 1934, 1955, 1936, 1980, 1966, 1949, 196
2,
       1964, 1952, 1933, 1942, 1939, 1954, 1945, 1961, 1965, 1938, 194
1,
       1931, 1937, 2022, 1932, 1923, 1915, 1928, 1922, 1917, 1913, 193
0,
       1926, 1914, 1924])
```

In [21]:
```
1 movie.head()
```

Out[21]:

	Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor
0	new_value	2019	NaN	Drama	NaN	NaN	J.S. Randhawa	Manmauji	Birb
1	Gadhvi (He thought he was Gandhi)	2019	109 min	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vive Ghamanc
2	Homecoming	2021	90 min	Drama, Musical	NaN	NaN	Soumyajit Majumdar	Sayani Gupta	Plabi Borthak
3	Yaaram	2019	110 min	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita R
4	...And Once Again	2010	105 min	Drama	NaN	NaN	Amol Palekar	Rajat Kapoor	Rituparr Sengup

In [22]:
```
1 movie['Duration'].unique()
```

Out[22]:
```
array([nan, '109 min', '90 min', '110 min', '105 min', '147 min',
       '142 min', '59 min', '82 min', '116 min', '96 min', '120 min',
       '161 min', '166 min', '102 min', '87 min', '132 min', '66 min',
       '146 min', '112 min', '168 min', '158 min', '126 min', '94 mi
n',
       '138 min', '124 min', '144 min', '157 min', '136 min', '107 mi
n',
       '113 min', '80 min', '122 min', '149 min', '148 min', '130 mi
n',
       '121 min', '188 min', '115 min', '103 min', '114 min', '170 mi
n',
       '100 min', '99 min', '140 min', '128 min', '93 min', '125 min',
       '145 min', '75 min', '111 min', '134 min', '85 min', '104 min',
       '92 min', '137 min', '127 min', '150 min', '119 min', '135 mi
n',
       '86 min', '76 min', '70 min', '72 min', '151 min', '95 min',
       '52 min', '89 min', '143 min', '177 min', '117 min', '123 min',
       '154 min', '88 min', '175 min', '153 min', '78 min', '139 min',
       '133 min', '101 min', '180 min', '60 min', '46 min', '164 min',
       '162 min', '171 min', '160 min', '152 min', '62 min', '163 mi
n',
       '165 min', '141 min', '210 min', '129 min', '156 min', '240 mi
n',
       '172 min', '155 min', '118 min', '167 min', '106 min', '193 mi
n',
       '57 min', '108 min', '45 min', '195 min', '174 min', '81 min',
       '178 min', '58 min', '184 min', '97 min', '98 min', '131 min',
       '176 min', '169 min', '77 min', '91 min', '84 min', '173 min',
       '74 min', '67 min', '181 min', '300 min', '79 min', '65 min',
       '48 min', '183 min', '159 min', '83 min', '68 min', '49 min',
       '201 min', '64 min', '186 min', '50 min', '69 min', '207 min',
       '55 min', '61 min', '185 min', '187 min', '216 min', '63 min',
       '54 min', '198 min', '51 min', '71 min', '73 min', '218 min',
       '191 min', '321 min', '199 min', '53 min', '56 min', '179 min',
       '47 min', '206 min', '190 min', '211 min', '247 min', '213 mi
n',
       '223 min', '2 min', '189 min', '224 min', '202 min', '255 min',
       '197 min', '182 min', '214 min', '208 min', '21 min', '200 mi
n',
       '192 min', '37 min', '261 min', '238 min', '204 min', '235 mi
n',
       '298 min', '217 min', '250 min'], dtype=object)
```

In [23]:
```
1 movie['Duration'].mode()
```

Out[23]:
```
0    120 min
Name: Duration, dtype: object
```

In [24]:
```
1 movie['Duration'].fillna('120 min',inplace=True)
```

In [25]:
```
1 movie['Duration'].unique()
```

Out[25]:
```
array(['120 min', '109 min', '90 min', '110 min', '105 min', '147 mi
n',
       '142 min', '59 min', '82 min', '116 min', '96 min', '161 min',
       '166 min', '102 min', '87 min', '132 min', '66 min', '146 min',
       '112 min', '168 min', '158 min', '126 min', '94 min', '138 mi
n',
       '124 min', '144 min', '157 min', '136 min', '107 min', '113 mi
n',
       '80 min', '122 min', '149 min', '148 min', '130 min', '121 mi
n',
       '188 min', '115 min', '103 min', '114 min', '170 min', '100 mi
n',
       '99 min', '140 min', '128 min', '93 min', '125 min', '145 min',
       '75 min', '111 min', '134 min', '85 min', '104 min', '92 min',
       '137 min', '127 min', '150 min', '119 min', '135 min', '86 mi
n',
       '76 min', '70 min', '72 min', '151 min', '95 min', '52 min',
       '89 min', '143 min', '177 min', '117 min', '123 min', '154 mi
n',
       '88 min', '175 min', '153 min', '78 min', '139 min', '133 min',
       '101 min', '180 min', '60 min', '46 min', '164 min', '162 min',
       '171 min', '160 min', '152 min', '62 min', '163 min', '165 mi
n',
       '141 min', '210 min', '129 min', '156 min', '240 min', '172 mi
n',
       '155 min', '118 min', '167 min', '106 min', '193 min', '57 mi
n',
       '108 min', '45 min', '195 min', '174 min', '81 min', '178 min',
       '58 min', '184 min', '97 min', '98 min', '131 min', '176 min',
       '169 min', '77 min', '91 min', '84 min', '173 min', '74 min',
       '67 min', '181 min', '300 min', '79 min', '65 min', '48 min',
       '183 min', '159 min', '83 min', '68 min', '49 min', '201 min',
       '64 min', '186 min', '50 min', '69 min', '207 min', '55 min',
       '61 min', '185 min', '187 min', '216 min', '63 min', '54 min',
       '198 min', '51 min', '71 min', '73 min', '218 min', '191 min',
       '321 min', '199 min', '53 min', '56 min', '179 min', '47 min',
       '206 min', '190 min', '211 min', '247 min', '213 min', '223 mi
n',
       '2 min', '189 min', '224 min', '202 min', '255 min', '197 min',
       '182 min', '214 min', '208 min', '21 min', '200 min', '192 mi
n',
       '37 min', '261 min', '238 min', '204 min', '235 min', '298 mi
n',
       '217 min', '250 min'], dtype=object)
```

In [26]:
```
1 movie['Genre'].unique()
```

```
       'Drama, Fantasy, Horror', 'Action, Adventure, Sci-Fi',
       'Drama, Action, Musical', 'Drama, Musical, Sport',
       'Action, Comedy, Horror', 'Drama, Fantasy, Musical',
       'Action, Fantasy, Musical', 'Animation, Action', 'Comedy, Mu
sic',
       'Documentary, Drama, Romance', 'Drama, Music, Thriller',
       'Fantasy, Musical, Mystery', 'Drama, Fantasy, War', 'Action,
War',
       'Action, Adventure, War', 'Horror, Musical',
       'Fantasy, Mystery, Thriller', 'Adventure, Biography, Drama',
       'Family, Romance, Sci-Fi', 'Drama, Romance, Family',
       'Animation, Adventure, Drama', 'Family, Romance, Drama',
       'Animation, Action, Sci-Fi', 'Adventure, Comedy, Fantasy',
       'Comedy, Crime, Family', 'Horror, Musical, Thriller',
       'Biography, Drama, Thriller', 'Drama, Western',
       'Romance, Sci-Fi, Thriller', 'Comedy, Musical, Family',
       'Comedy, Horror, Romance', 'Thriller, Action',
       'Fantasy, Thriller, Action', 'Fantasy, Romance',
       'Action, Drama, Comedy', 'Family, Fantasy, Romance',
```

In [27]:
```
1 movie['Genre'].mode()
```

Out[27]:
```
0    Drama
Name: Genre, dtype: object
```

In [28]:
```
1 movie['Genre'].fillna('Drama',inplace=True)
```

In [29]:
```
1 movie.head()
```

Out[29]:

	Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor
0	new_value	2019	120 min	Drama	NaN	NaN	J.S. Randhawa	Manmauji	Birb
1	Gadhvi (He thought he was Gandhi)	2019	109 min	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vive Ghamanc
2	Homecoming	2021	90 min	Drama, Musical	NaN	NaN	Soumyajit Majumdar	Sayani Gupta	Plabi Borthak
3	Yaaram	2019	110 min	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita R
4	...And Once Again	2010	105 min	Drama	NaN	NaN	Amol Palekar	Rajat Kapoor	Rituparr Sengup

In [30]:
```
1 movie['Rating'].mean()
```

Out[30]:
```
5.841621416845562
```

In [31]:
```
1 movie['Rating']=pd.to_numeric(movie['Rating'],errors='coerce').asty
```

In [32]:
```
movie['Rating']=movie['Rating'].astype(float)
```

In [33]:
```
movie['Rating'].fillna('5.8',inplace=True)
```

In [34]:
```
movie['Votes'].unique()
```

Out[34]:
```
array([nan, '8', '35', ..., '70,344', '408', '1,496'], dtype=object)
```

In [35]:
```
movie['Votes'] = pd.to_numeric(movie['Votes'], errors='coerce').fil
```

In [36]:
```
movie['Votes'].mean()
```

Out[36]:
```
51.01921464955832
```

In [37]:
```
movie['Votes'].replace(to_replace=['0','51'], inplace=True)
```

In [38]:
```
movie['Votes'].dtype
```

Out[38]:
```
dtype('int32')
```

In [39]:
```
movie['Votes'].replace(to_replace='0', value='51', inplace=True)
```

In [40]:
```
replace_values={'0':'51'}
movie['Votes']=movie['Votes'].map(replace_values).fillna(movie['Vot
```

In [41]:
```
movie['Votes'].replace(0, 51, inplace=True)
```

In [42]:
```
movie.head()
```

Out[42]:

	Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor
0	new_value	2019	120 min	Drama	5.8	51	J.S. Randhawa	Manmauji	Birb
1	Gadhvi (He thought he was Gandhi)	2019	109 min	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vive Ghamanc
2	Homecoming	2021	90 min	Drama, Musical	5.8	51	Soumyajit Majumdar	Sayani Gupta	Plabi Borthak
3	Yaaram	2019	110 min	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita R
4	...And Once Again	2010	105 min	Drama	5.8	51	Amol Palekar	Rajat Kapoor	Rituparr Sengup

In [43]:
```
movie['Director'].nunique()
```

Out[43]:
```
5938
```

In [44]:
```
movie['Director'].fillna('Jayant Desai',inplace=True)
```

In [45]:
```
movie['Director'].mode()
```

Out[45]:
```
0    Jayant Desai
Name: Director, dtype: object
```

In [46]:
```
movie['Actor 1'].unique()
```

Out[46]:
```
array(['Manmauji', 'Rasika Dugal', 'Sayani Gupta', ..., 'Meghan Jadha
v',
       'Roohi Berde', 'Sangeeta Tiwari'], dtype=object)
```

In [47]:
```
movie['Actor 1'].mode()
```

Out[47]:
```
0    Ashok Kumar
Name: Actor 1, dtype: object
```

In [48]:
```
movie['Actor 1'].fillna('Ashok Kumar',inplace=True)
```

In [49]:
```
movie['Actor 1'].replace(to_replace=[', ..., ','Ashok Kumar'], inpl
```

In [50]:
```
movie['Actor 2'].unique()
```

Out[50]:
```
array(['Birbal', 'Vivek Ghamande', 'Plabita Borthakur', ...,
       'Devan Sanjeev', 'Prince Daniel', 'Sarah Jane Dias'], dtype=obj
ect)
```

In [51]:
```
movie['Actor 2'].mode()
```

Out[51]:
```
0    Rekha
Name: Actor 2, dtype: object
```

In [52]:
```
movie['Actor 2'].fillna('Rekha',inplace=True)
```

In [53]:
```
movie['Actor 2'].replace(to_replace=[' ...','Rekha'],inplace=True)
```

In [54]:
```
movie['Actor 2'].replace(' ...','Rekha', inplace=True)
```

In [55]:
```
movie['Actor 3'].unique()
```

Out[55]:
```
array(['Rajendra Bhatia', 'Arvind Jangid', 'Roy Angana', ...,
       'Shatakshi Gupta', 'Valerie Agha', 'Suparna Anand'], dtype=obje
ct)
```

In [56]:
```
movie['Actor 3'].mode()
```

Out[56]:
```
0    Pran
Name: Actor 3, dtype: object
```

In [57]:
```
movie['Actor 3'].fillna('Pran',inplace=True)
```

In [58]:
```
movie['Actor 3'].replace(' ...','Pran',inplace=True)
```

In [59]:
```
movie.isnull().sum()
```

Out[59]:
```
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 [60]:
```
movie.head()
```

Out[60]:

	Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor
0	new_value	2019	120 min	Drama	5.8	51	J.S. Randhawa	Manmauji	Birb
1	Gadhvi (He thought he was Gandhi)	2019	109 min	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vive Ghamanc
2	Homecoming	2021	90 min	Drama, Musical	5.8	51	Soumyajit Majumdar	Sayani Gupta	Plabi Borthak
3	Yaaram	2019	110 min	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita R
4	...And Once Again	2010	105 min	Drama	5.8	51	Amol Palekar	Rajat Kapoor	Rituparr Sengup

In [61]:
```
movie.describe()
```

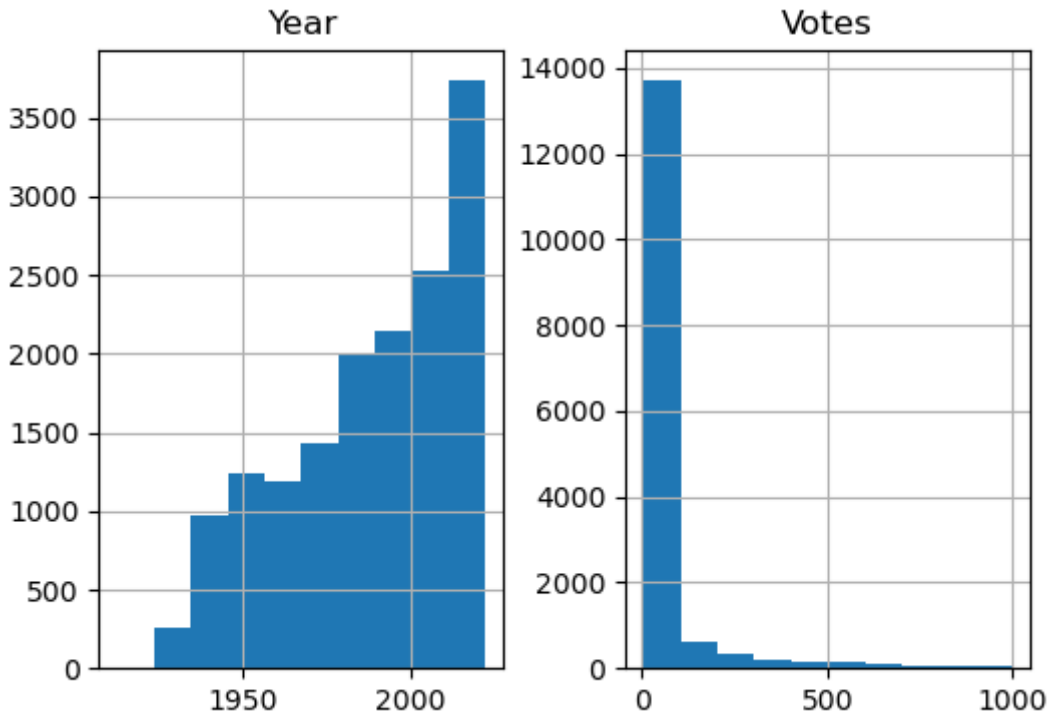
Out[61]:

	Year	Votes
count	15509.000000	15509.000000
mean	1988.101232	80.486685
std	25.644978	130.818012
min	1913.000000	5.000000
25%	1969.000000	51.000000
50%	1992.000000	51.000000
75%	2011.000000	51.000000
max	2022.000000	999.000000

In [62]:
```
ratings = movie["Rating"].value_counts()
numbers = ratings.index
quantity = ratings.values
import plotly.express as px
fig = px.pie(movie, values=quantity, names=numbers)
```

In [63]:
```
movie.hist(figsize=[6,4])
```

Out[63]:
```
array([[<Axes: title={'center': 'Year'}>,
        <Axes: title={'center': 'Votes'}>]], dtype=object)
```



Build a model that predicts the rating of a movie based on features like genre, director, and actors. You can use regression techniques to tackle this problem.

In [64]:
```
from sklearn.preprocessing import LabelEncoder
```

In [65]:
```
# Encoding categorical variables
label_encoders = {}
categorical_columns = ['Genre', 'Director', 'Actor 1', 'Actor 2', '
for column in categorical_columns:
    label_encoders[column] = LabelEncoder()
    movie[column] = label_encoders[column].fit_transform(movie[colu
```

In [66]:
```
from sklearn.linear_model import LinearRegression
```

In [67]:
```
X = movie[['Genre', 'Director', 'Actor 1', 'Actor 2', 'Actor 3']]
y = movie['Rating']
```

In [68]:
```
#training
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,ra
```

In [69]:
```
model=LinearRegression()
model.fit(X_train,y_train)
```

Out[69]:
```
▾ LinearRegression

LinearRegression()
```

In [70]:
```
from sklearn.metrics import mean_squared_error
```

In [71]:
```
#prediction
y_prediction = model.predict(X_test)
y_prediction
```

Out[71]:
```
array([5.79327853, 5.75706241, 5.78126597, ..., 5.8068149 , 5.8316189
8,
       5.65776968])
```

In [72]:
```
prediction= mean_squared_error(y_test, y_prediction)
```

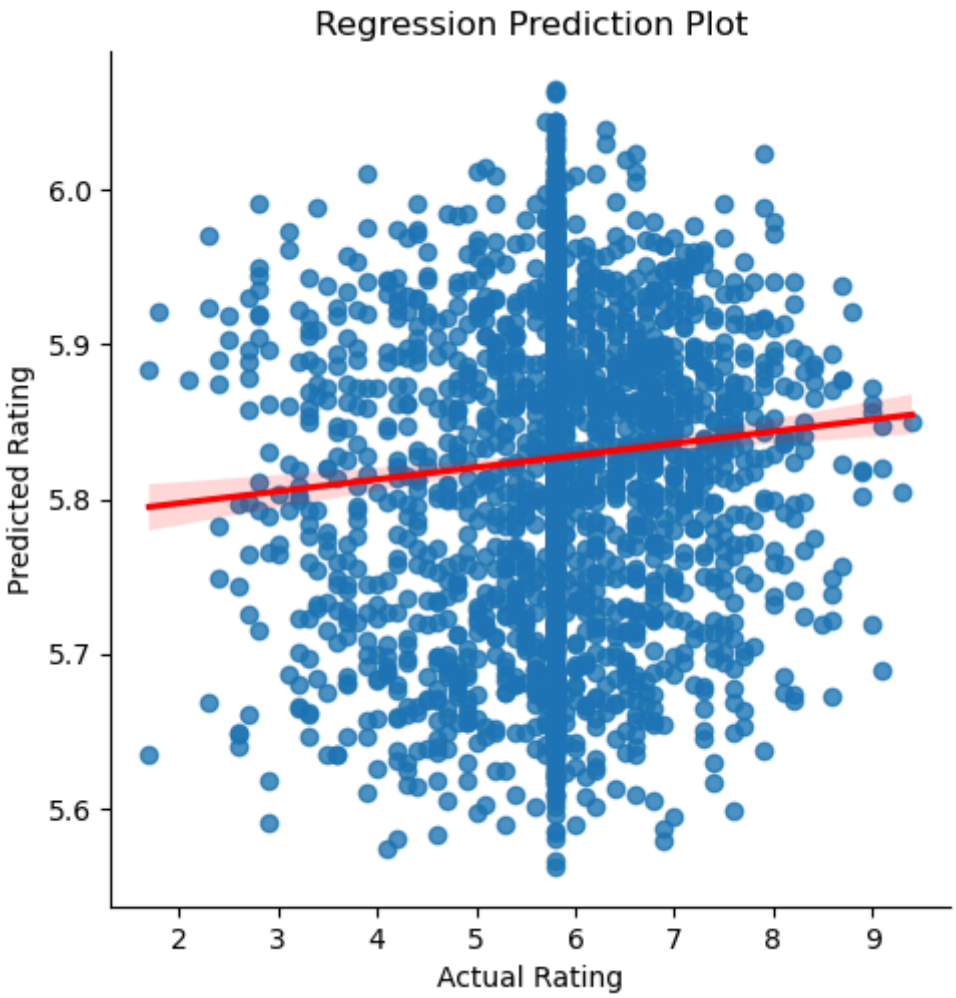
In [73]:
```
#scatter plot
import matplotlib.pyplot as plt
import seaborn as sns
```

In [74]:

```
 # Convert numerical values to numeric data type
y_test = pd.to_numeric(y_test)
y_pred = pd.to_numeric(y_prediction)
# DataFrame for actual and predicted values
results = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
sns.lmplot(x='Actual', y='Predicted', data=results, line_kws={'colo
plt.xlabel('Actual Rating')
plt.ylabel('Predicted Rating')
plt.title('Regression Prediction Plot')
print('Mean Squared Error :',prediction)
plt.show()

```

```
Mean Squared Error : 0.9403051683042151
```



In [75]:

```
print('Mean Squared Error :',prediction)
```

```
Mean Squared Error : 0.9403051683042151
```

In [76]:

```
from sklearn.metrics import r2_score
```

In [77]:

```
r2=r2_score(y_test,y_prediction)
```

In [78]:
```
1 print('r2_error is ',r2)
```

```
r2_error is  0.0057301218238191565
```

Conclusion

Mainly we focused on Linear Regression

We took movie rating dataset and performed a linear regression algorithm

Finally, An R-squared value of 0.90 means that 90% of the variance in the dependent variable (sales, in this case) can be explained by the independent variables (TV, Radio, Newspaper advertising expenditures).

In [ ]:
```
1
```