In [1]:
```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
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
from sklearn.metrics import mean_squared_error,mean_absolute_error
%matplotlib inline

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [3]:
```
df = pd.read_csv('C:/Users/HP/Downloads/IMDb Movies India.csv', encoding='ISO-8859-
```

In [4]:
```
df.head()
```

Out[4]:

	ï»¿Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Acto
0		NaN	NaN	Drama	NaN	NaN	J.S. Randhawa	Manmauji	Bi
1	#Gadhvi (He thought he was Gandhi)	-2019.0	109 min	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vi Ghama
2	#Homecoming	-2021.0	90 min	Drama, Musical	NaN	NaN	Soumyajit Majumdar	Sayani Gupta	Pla Bortha
3	#Yaaram	-2019.0	110 min	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita
4	...And Once Again	-2010.0	105 min	Drama	NaN	NaN	Amol Palekar	Rajat Kapoor	Ritupa Sengu

In [5]:
```
df.columns
```

Out[5]:
```
Index(['ï»¿Name', 'Year', 'Duration', 'Genre', 'Rating', 'Votes', 'Director',
       'Actor 1', 'Actor 2', 'Actor 3'],
      dtype='object')
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15509 entries, 0 to 15508
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   ï»¿Name   15509 non-null  object 
 1   Year      14981 non-null  float64
 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(2), object(8)
memory usage: 1.2+ MB
```

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

Out[7]:

	Year	Rating
count	14981.000000	7919.000000
mean	-1987.012215	5.841621
std	25.416689	1.381777
min	-2022.000000	1.100000
25%	-2009.000000	4.900000
50%	-1991.000000	6.000000
75%	-1968.000000	6.800000
max	-1913.000000	10.000000

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
ï»¿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 [9]:
```
df.dropna(inplace=True)
df.isnull().sum()
```

Out[9]:
```
ï»¿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 [10]:
```
df.head()
```

Out[10]:

	ï»¿Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor 2	A
1	#Gadhvi (He thought he was Gandhi)	-2019.0	109 min	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vivek Ghamande	
3	#Yaaram	-2019.0	110 min	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita Raj	Sid K
5	...Aur Pyaar Ho Gaya	-1997.0	147 min	Comedy, Drama, Musical	4.7	827	Rahul Rawail	Bobby Deol	Aishwarya Rai Bachchan	Sh K
6	...Yahaan	-2005.0	142 min	Drama, Romance, War	7.4	1,086	Shoojit Sircar	Jimmy Sheirgill	Minissha Lamba	Y S
8	?: A Question Mark	-2012.0	82 min	Horror, Mystery, Thriller	5.6	326	Allyson Patel	Yash Dave	Muntazir Ahmad	

In [14]:
```
# Remove non-numeric characters from the 'Year' column
df['Year'] = df['Year'].astype(str).str.replace(r'[^\d]', '', regex=True)

# Convert the 'Year' column to integers
df['Year'] = df['Year'].astype(int)
```

In [15]:
```
df['Duration'] = df['Duration'].str.strip('min')
df.info()
df.head()
```

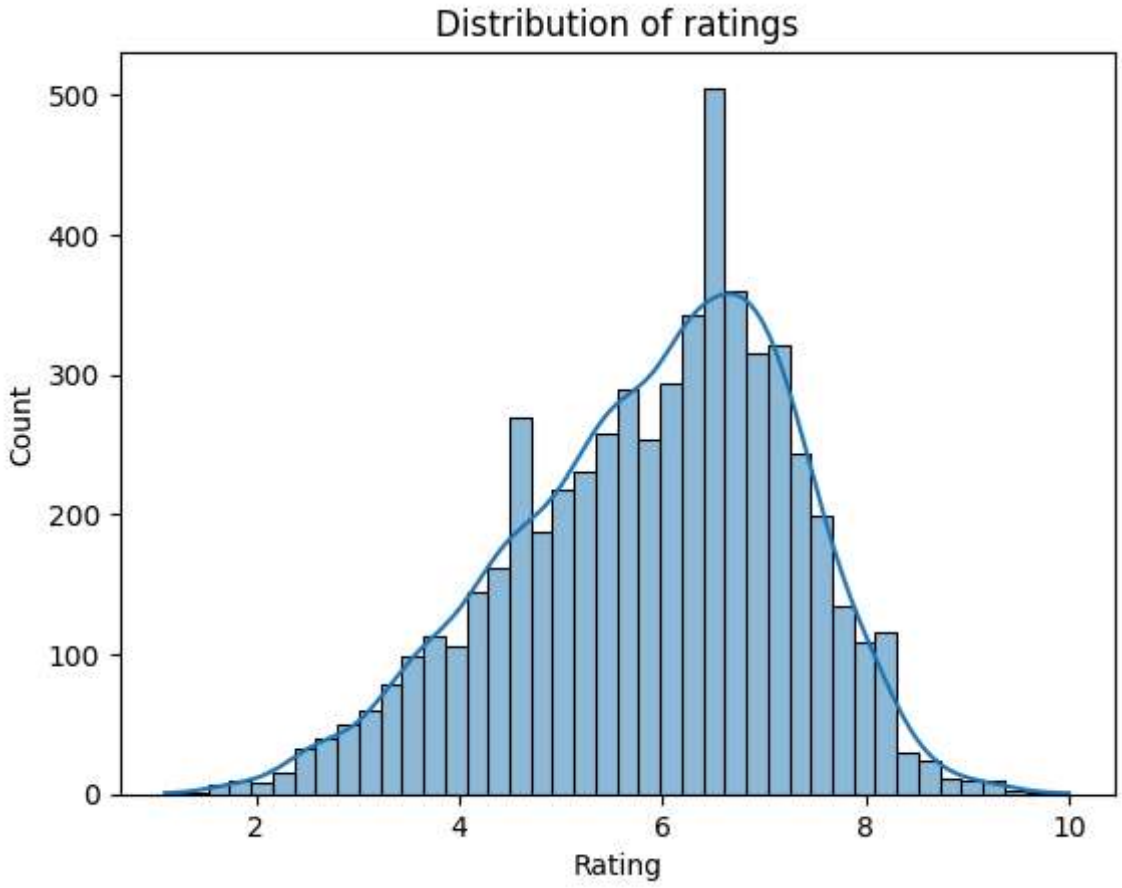
```
<class 'pandas.core.frame.DataFrame'>
Index: 5659 entries, 1 to 15508
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   ï»¿Name   5659 non-null   object
 1   Year      5659 non-null   int32
 2   Duration  5659 non-null   object
 3   Genre     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
```

Out[15]:

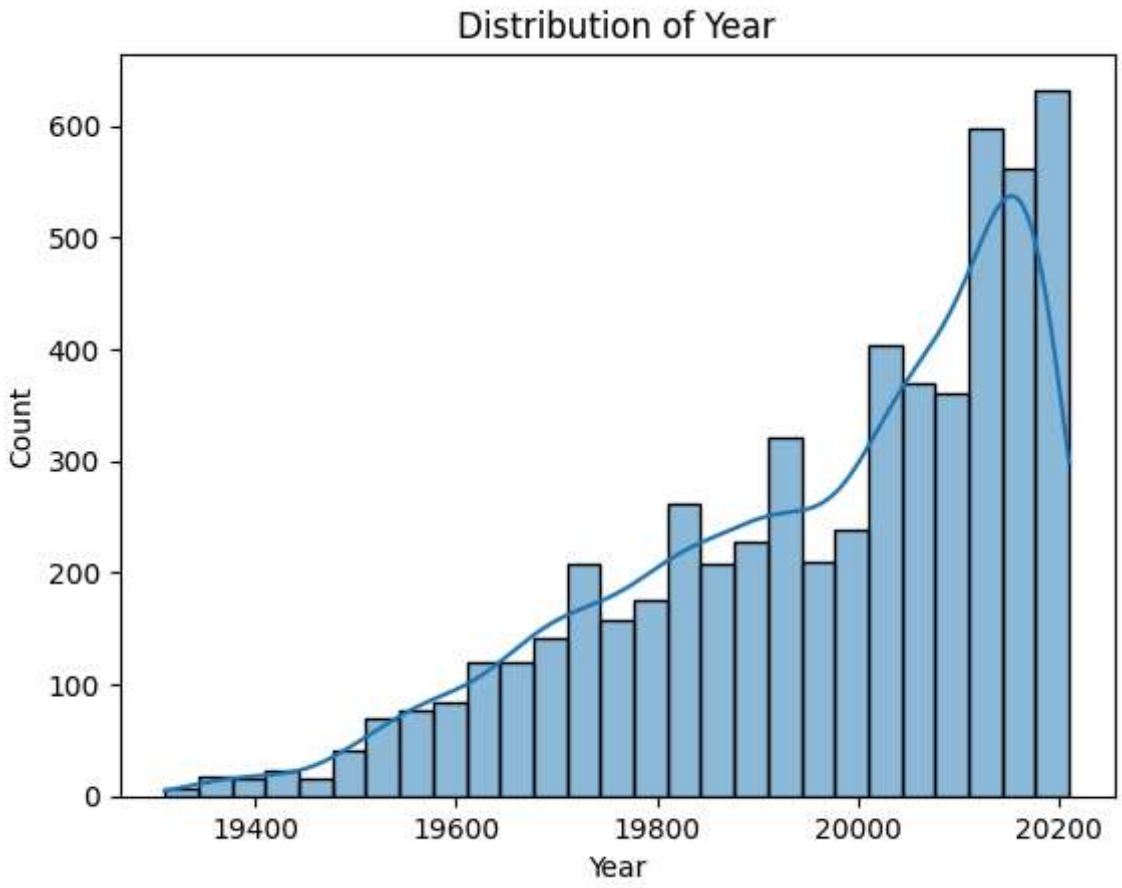
	ï»¿Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor 2	Act
1	#Gadhvi (He thought he was Gandhi)	20190	109	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vivek Ghamande	Ar Ja
3	#Yaaram	20190	110	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita Raj	Sidd Ka
5	...Aur Pyaar Ho Gaya	19970	147	Comedy, Drama, Musical	4.7	827	Rahul Rawail	Bobby Deol	Aishwarya Rai Bachchan	Sha Ka
6	...Yahaan	20050	142	Drama, Romance, War	7.4	1086	Shoojit Sircar	Jimmy Sheirgill	Minissha Lamba	Yas Sha
8	?: A Question Mark	20120	82	Horror, Mystery, Thriller	5.6	326	Allyson Patel	Yash Dave	Muntazir Ahmad	I B

In [16]:

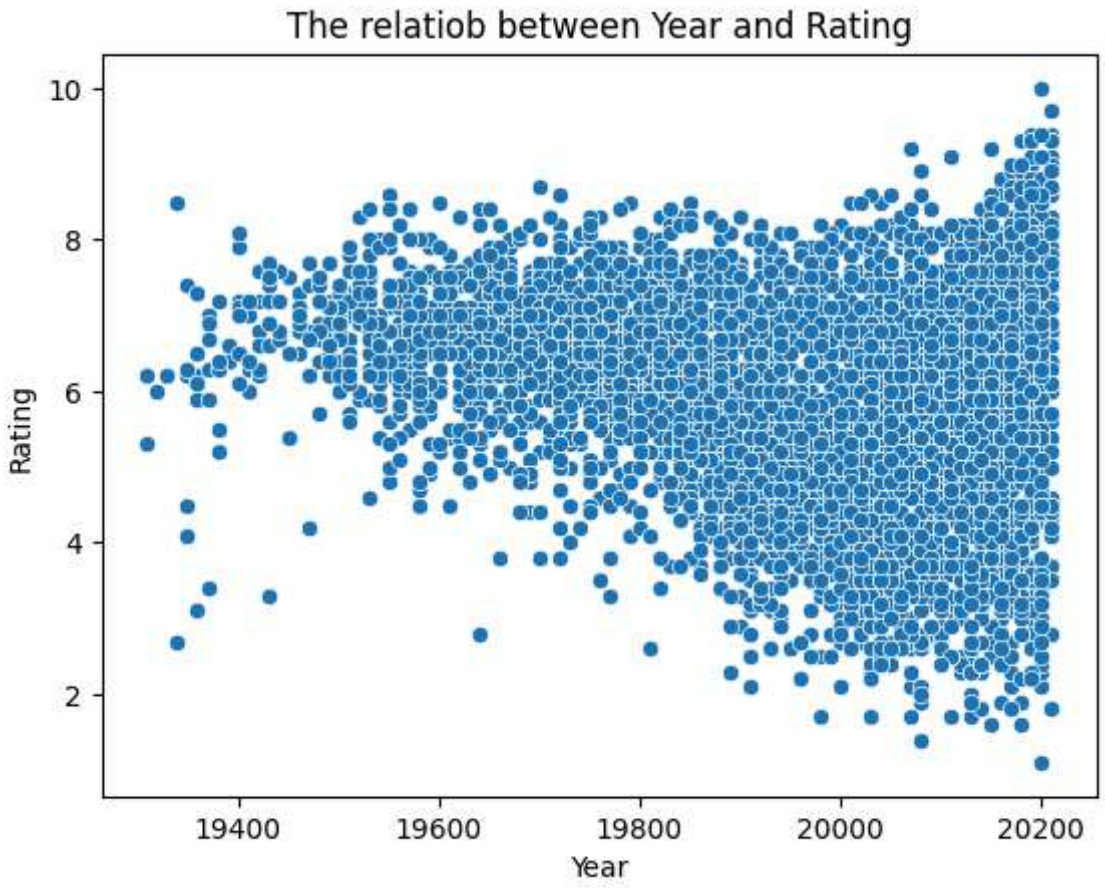
```
sns.histplot(data=df,x='Rating',kde=True)
plt.title('Distribution of ratings')
plt.show()
```



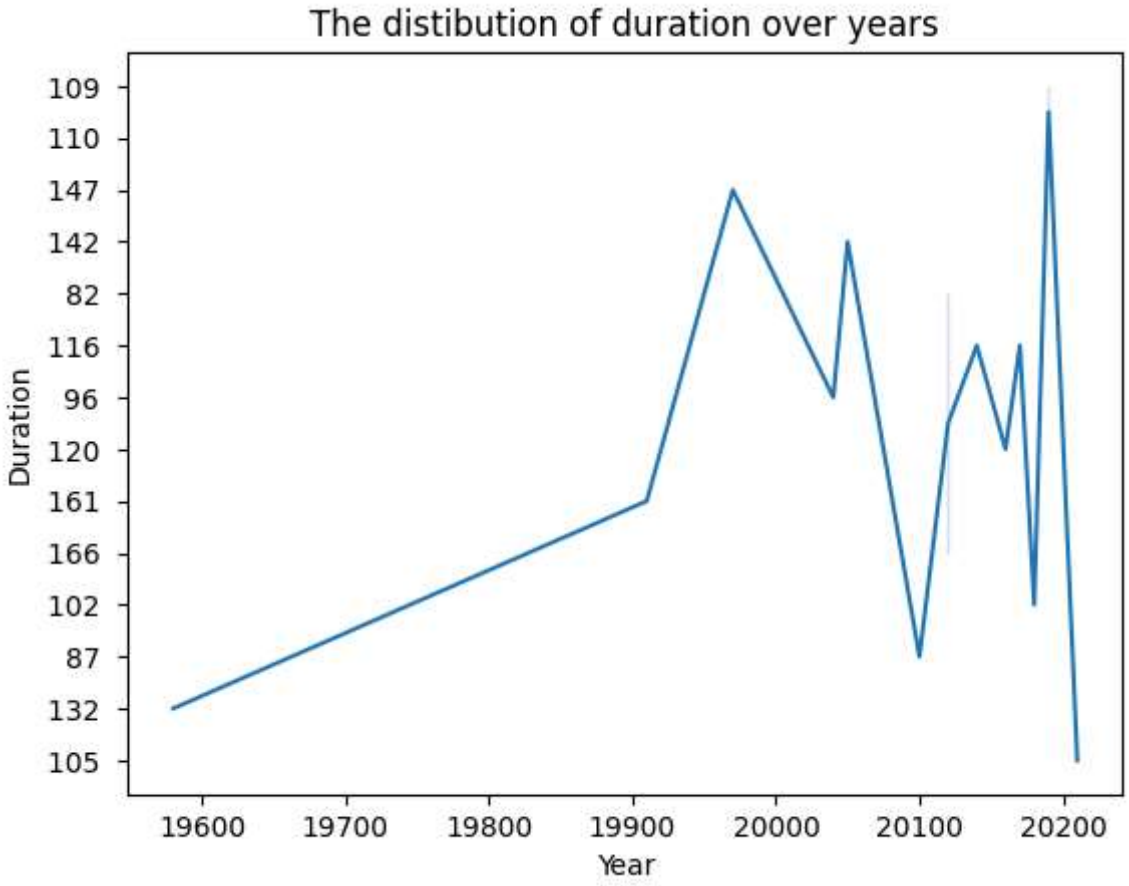
In [17]:
```
sns.histplot(data=df,x='Year',kde=True)
plt.title('Distribution of Year')
plt.show()
```



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

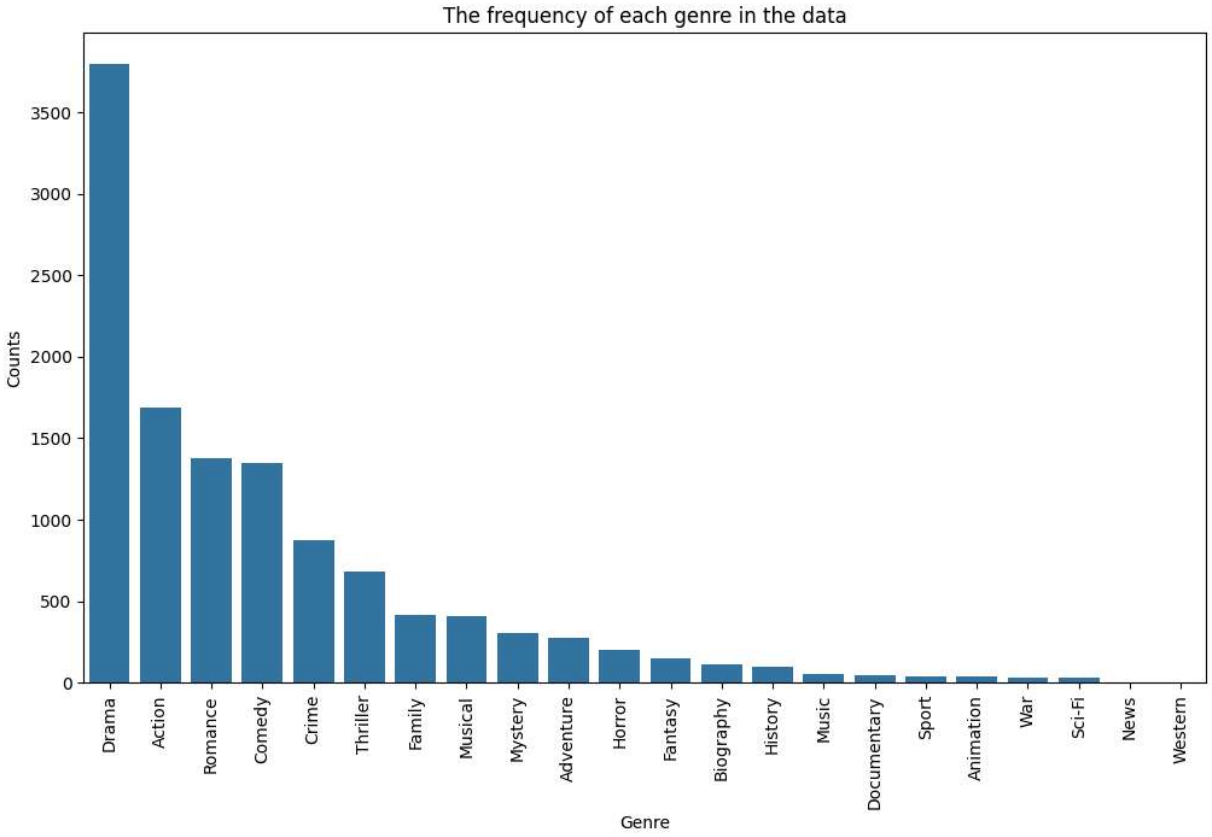


In [19]:
```
sns.lineplot(data=df.head(15),x='Year',y='Duration')
plt.title('The distibution of duration over years')
plt.show()
```



In [20]:
```
movies_genre = df['Genre'].str.split(', ',expand=True).stack().value_counts()
labels = movies_genre.keys()
count = movies_genre.values
print(movies_genre)
print(labels)
print(count)
plt.figure(figsize=(12,7))
sns.barplot(x=labels,y=count)
plt.xticks(rotation=90)
plt.title('The frequency of each genre in the data')
plt.xlabel('Genre')
plt.ylabel('Counts')
plt.show()
```

```
Drama          3796
Action         1686
Romance        1380
Comedy         1344
Crime           875
Thriller        679
Family          416
Musical         412
Mystery         304
Adventure       277
Horror          202
Fantasy         146
Biography       115
History          99
Music            53
Documentary      48
Sport            40
Animation        40
War              33
Sci-Fi           32
News              1
Western           1
Name: count, dtype: int64
Index(['Drama', 'Action', 'Romance', 'Comedy', 'Crime', 'Thriller', 'Family',
       'Musical', 'Mystery', 'Adventure', 'Horror', 'Fantasy', 'Biography',
       'History', 'Music', 'Documentary', 'Sport', 'Animation', 'War',
       'Sci-Fi', 'News', 'Western'],
      dtype='object')
[3796 1686 1380 1344  875  679  416  412  304  277  202  146  115   99
   53   48   40   40   33   32    1    1]
```



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

```
<class 'pandas.core.frame.DataFrame'>
Index: 5659 entries, 1 to 15508
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   ï»¿Name   5659 non-null   object 
 1   Year      5659 non-null   int32  
 2   Duration  5659 non-null   object 
 3   Genre     5659 non-null   int32  
 4   Rating    5659 non-null   float64
 5   Votes     5659 non-null   int32  
 6   Director  5659 non-null   int32  
 7   Actor 1   5659 non-null   int32  
 8   Actor 2   5659 non-null   int32  
 9   Actor 3   5659 non-null   int32  
dtypes: float64(1), int32(7), object(2)
memory usage: 460.6+ KB
```

In [22]:
```
df.head()
```

Out[22]:

	ï»¿Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor 2	Actor 3
1	#Gadhvi (He thought he was Gandhi)	20190	109	229	7.0	8	629	1352	2272	319
3	#Yaaram	20190	110	184	4.4	35	1335	1198	719	2148
5	...Aur Pyaar Ho Gaya	19970	147	157	4.7	827	1530	378	75	2045
6	...Yahaan	20050	142	289	7.4	1086	2044	692	1112	2524
8	?: A Question Mark	20120	82	320	5.6	326	135	1934	1175	1013

In [28]:
```
print(df.columns)
```

```
Index(['ï»¿Name', 'Year', 'Duration', 'Genre', 'Rating', 'Votes', 'Director',
       'Actor 1', 'Actor 2', 'Actor 3'],
      dtype='object')
```

In [31]:
```
# Assuming df is already defined and contains the necessary columns
df2 = df.copy()
```

```
# Splitting data into features (X) and target variable (y)
X = df2.drop('Rating', axis=1)
y = df2['Rating']

# Splitting data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta
```

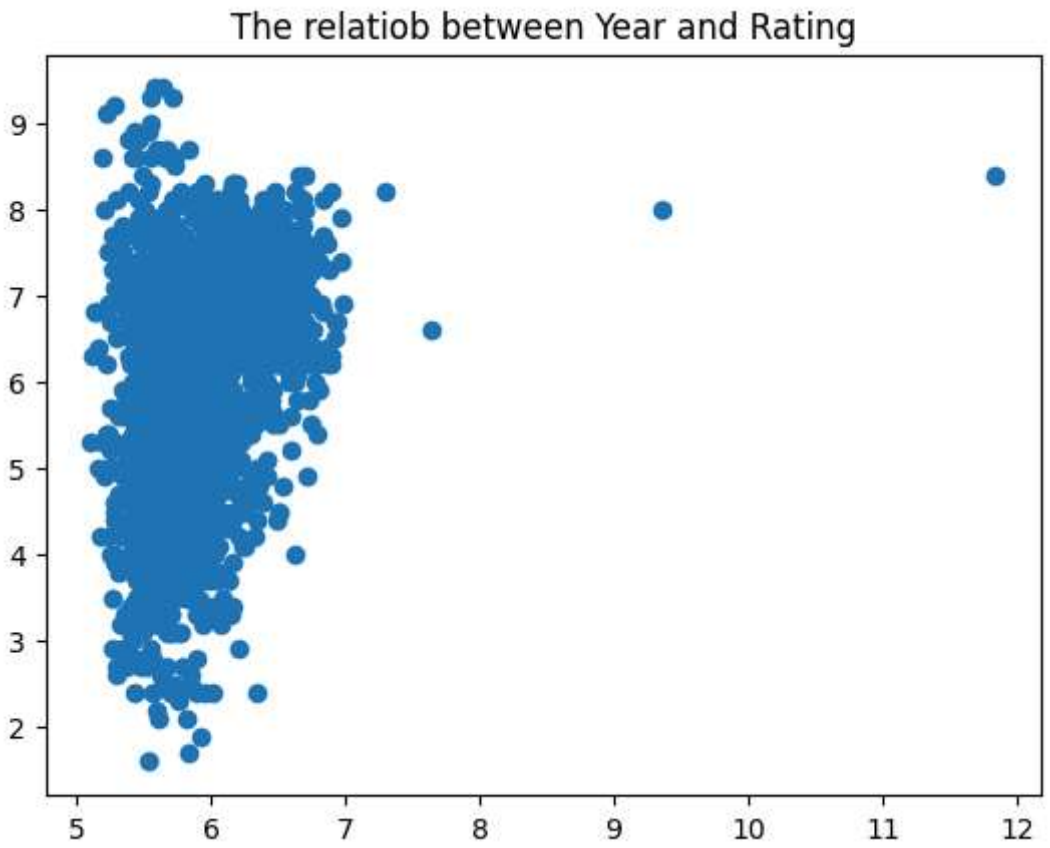
In [33]:
```
# Exclude non-numeric columns from features
numeric_columns = X.select_dtypes(include=['number']).columns
X_numeric = X[numeric_columns]

# Splitting data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_numeric, y, test_size=0.3, ra

# Training the model
model = LinearRegression()
model.fit(X_train, y_train)

# Predicting on the test set
y_pred = model.predict(X_test)
```

In [34]:
```
# sns.scatterplot(x='y_pred',y='y_test')
plt.scatter(y_pred,y_test)
plt.title("The relatiob between Year and Rating")
plt.show()
```



In [35]:
```
print(f"Mean Absolute Error : {mean_absolute_error(y_test,y_pred)}")
print(f"Mean Squared Error : {mean_squared_error(y_test,y_pred)}")
```

```
Mean Absolute Error : 1.0367425303764857
Mean Squared Error : 1.6770401510539197
```

In [ ]: