import pandas as pd import numpy as np movies=pd.read_csv(r'C:\Venkat\Python\Practice_Material\Kaggle_DataSets\movie.csv',sep=',') #movies

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
movies=pd.read_csv(r'C:\Venkat\Python\Practice_Material\Kaggle_DataSets\movie.csv',sep=',')
#movies
```

movies.head(20)

In [6]:
```
movies.head(20)
```

Out[6]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy
5	6	Heat (1995)	Action|Crime|Thriller
6	7	Sabrina (1995)	Comedy|Romance
7	8	Tom and Huck (1995)	Adventure|Children
8	9	Sudden Death (1995)	Action
9	10	GoldenEye (1995)	Action|Adventure|Thriller
10	11	American President, The (1995)	Comedy|Drama|Romance
11	12	Dracula: Dead and Loving It (1995)	Comedy|Horror
12	13	Balto (1995)	Adventure|Animation|Children
13	14	Nixon (1995)	Drama
14	15	Cutthroat Island (1995)	Action|Adventure|Romance
15	16	Casino (1995)	Crime|Drama
16	17	Sense and Sensibility (1995)	Drama|Romance
17	18	Four Rooms (1995)	Comedy
18	19	Ace Ventura: When Nature Calls (1995)	Comedy
19	20	Money Train (1995)	Action|Comedy|Crime|Drama|Thriller

In [101…
```
tags=pd.read_csv(r'C:\Venkat\Python\Practice_Material\Kaggle_DataSets\tag.csv',sep=',')
tags.head()
```

Out[101…

	userId	movieId	tag	timestamp
0	18	4141	Mark Waters	2009-04-24 18:19:40
1	65	208	dark hero	2013-05-10 01:41:18
2	65	353	dark hero	2013-05-10 01:41:19
3	65	521	noir thriller	2013-05-10 01:39:43
4	65	592	dark hero	2013-05-10 01:41:18

In [102…
```
ratings=pd.read_csv(r'C:\Venkat\Python\Practice_Material\Kaggle_DataSets\rating.csv',sep=',')
ratings.head()
```

Out[102…

	userId	movieId	rating	timestamp
0	1	2	3.5	2005-04-02 23:53:47
1	1	29	3.5	2005-04-02 23:31:16
2	1	32	3.5	2005-04-02 23:33:39
3	1	47	3.5	2005-04-02 23:32:07
4	1	50	3.5	2005-04-02 23:29:40

In [7]:
```
del ratings['timestamp']
del tags['timestamp']
```

In [14]:
```
ratings.head()
```

Out[14]:

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5
2	1	32	3.5
3	1	47	3.5
4	1	50	3.5

In [9]:
```
tags.columns
```

Out[9]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [14]:
```
#Sereies
row_0=tags.iloc[0]
print(row_0)
```

```
userId              18
movieId           4141
tag        Mark Waters
Name: 0, dtype: object
```

In [48]:
```
row_0['userId']
```

Out[48]:
```
18
```

In [50]:
```
row_0.index
```

Out[50]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [52]:
```
row_0.info
```

Out[52]:
```
<bound method Series.info of userId              18
movieId           4141
tag        Mark Waters
Name: 0, dtype: object>
```

In [54]:
```
'rating' in row_0
r
```

Out[54]:
```
False
```

In [56]:
```
row_0.name
```

Out[56]:
```
0
```

In [58]:
```
row_0.name
```

Out[58]:
```
0
```

In [60]:
```
row_0=row_0.rename('firstrow')
```

In [62]:
```
row_0.name
```

Out[62]:
```
'firstrow'
```

In [64]:
```
tags.index
```

Out[64]:
```
RangeIndex(start=0, stop=465564, step=1)
```

In [66]:
```
tags.columns
```

Out[66]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [70]:
```
tags.iloc[[0,11,500]]
```

Out[70]:

	userId	movieId	tag
0	18	4141	Mark Waters
11	65	1783	noir thriller
500	342	55908	entirely dialogue

In [74]:
```
ratings.head(20)
```

Out[74]:

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5
2	1	32	3.5
3	1	47	3.5
4	1	50	3.5
5	1	112	3.5
6	1	151	4.0
7	1	223	4.0
8	1	253	4.0
9	1	260	4.0
10	1	293	4.0
11	1	296	4.0
12	1	318	4.0
13	1	337	3.5
14	1	367	3.5
15	1	541	4.0
16	1	589	3.5
17	1	593	3.5
18	1	653	3.0
19	1	919	3.5

In [76]:
```
ratings['rating'].describe()
```

Out[76]:
```
count    2.000026e+07
mean     3.525529e+00
std      1.051989e+00
min      5.000000e-01
25%      3.000000e+00
50%      3.500000e+00
75%      4.000000e+00
max      5.000000e+00
Name: rating, dtype: float64
```

In [78]:
```
ratings.isnull
```

Out[78]:
```
<bound method DataFrame.isnull of           userId  movieId  rating
0              1        2     3.5
1              1       29     3.5
2              1       32     3.5
3              1       47     3.5
4              1       50     3.5
...          ...      ...     ...
20000258  138493    68954     4.5
20000259  138493    69526     4.5
20000260  138493    69644     3.0
20000261  138493    70286     5.0
20000262  138493    71619     2.5

[20000263 rows x 3 columns]>
```

In [80]:
```
ratings.isnull()
```

Out[80]:

	userId	movieId	rating
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
...	...	...	...
20000258	False	False	False
20000259	False	False	False
20000260	False	False	False
20000261	False	False	False
20000262	False	False	False

20000263 rows × 3 columns

In [82]:
```
ratings.isnull().any()
```

Out[82]:
```
userId     False
movieId    False
rating     False
dtype: bool
```

In [84]:
```
ratings.isnull().any().any()
```

Out[84]:
```
False
```

In [86]:
```
movies.isnull()
```

Out[86]:

	movieId	title	genres
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
...	...	...	...
27273	False	False	False
27274	False	False	False
27275	False	False	False
27276	False	False	False
27277	False	False	False

27278 rows × 3 columns

In [88]:
```
movies.isnull().any()
```

Out[88]:
```
movieId    False
title      False
genres     False
dtype: bool
```

In [90]:
```
movies.isnull().any().any()
```

Out[90]:
```
False
```

In [92]:
```
tags.isnull()
```

Out[92]:

	userId	movieId	tag
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
...	...	...	...
465559	False	False	False
465560	False	False	False
465561	False	False	False
465562	False	False	False
465563	False	False	False

465564 rows × 3 columns

In [94]:
```
tags.isnull().any()
```

Out[94]:
```
userId     False
movieId    False
tag         True
dtype: bool
```

In [96]:
```
tags.isnull().any().any()
```

Out[96]:
```
True
```

In [108…
```
tags[tags.isnull().all(axis=1)]
```

Out[108…

userId	movieId	tag

In [110…
```
tags=tags.dropna()
```

In [112…
```
tags.isnull().any().any()
```

Out[112…
```
False
```

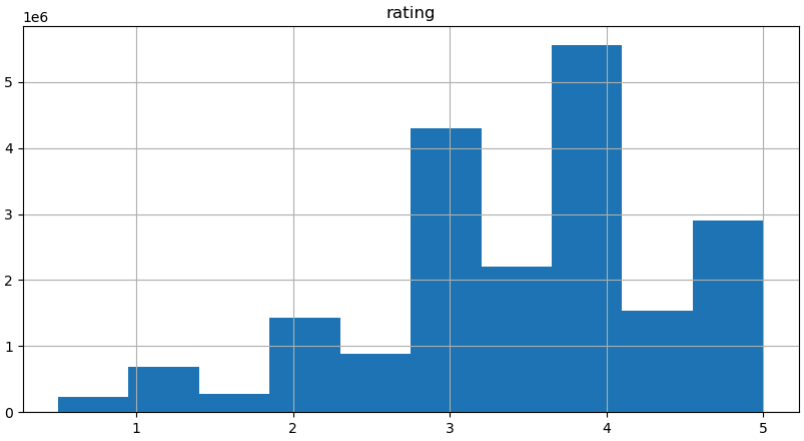
In [116…
```
ratings.head(5)
```

Out[116…

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5
2	1	32	3.5
3	1	47	3.5
4	1	50	3.5

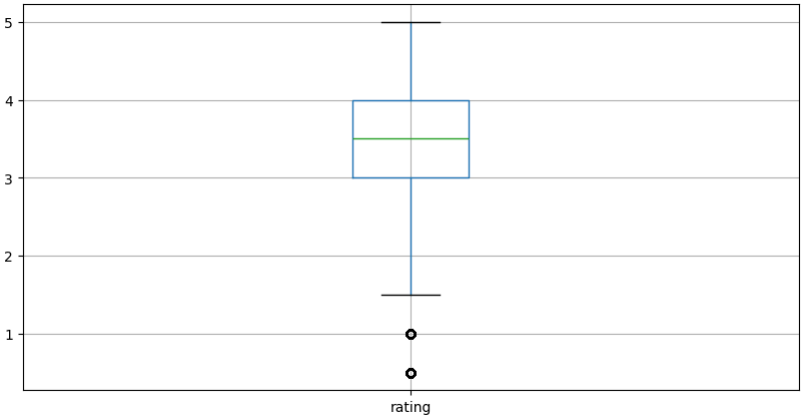
In [122…
```
ratings.shape
%matplotlib inline
ratings.hist(column='rating',figsize=(10,5))
```

Out[122…
```
array([[<Axes: title={'center': 'rating'}>]], dtype=object)
```



In [129…
```
%matplotlib inline
ratings.boxplot(column='rating',figsize=(10,5))
```

Out[129…
```
<Axes: >
```



In [18]:
```
ratings[ratings['rating']==5].sum()
```

Out[18]:
```
userId     2.003319e+11
movieId    1.819012e+10
rating     1.449330e+07
dtype: float64
```

In [153…
```
data = {
    'A': [1, 6, 3, 8, 5],
    'B': [2, 7, 3, 1, 9],
    'C': [5, 5, 5, 5, 5]
}
print(data)
#count_greater_than_5 = data[data['A'] > 5].sum()

#print(f"Count of rows in column 'A' greater than 5: {count_greater_than_5}")
```

```
{'A': [1, 6, 3, 8, 5], 'B': [2, 7, 3, 1, 9], 'C': [5, 5, 5, 5, 5]}
```

In [20]:
```
ratings['rating'].max()
```

Out[20]:
```
5.0
```

In [22]:
```
ratings['rating'].min
```

Out[22]:
```
<bound method Series.min of 0           3.5
1           3.5
2           3.5
3           3.5
4           3.5
           ...
20000258    4.5
20000259    4.5
20000260    3.0
20000261    5.0
20000262    2.5
Name: rating, Length: 20000263, dtype: float64>
```

In [24]:
```
ratings['rating'].min()
```

Out[24]:
```
0.5
```

In [32]:
```
type(ratings['rating'])
```

Out[32]:
```
pandas.core.series.Series
```

In [40]:
```
ratings[ratings['rating']==0.5].count()
```

Out[40]:
```
userId     239125
movieId    239125
rating     239125
dtype: int64
```

In [42]:
```
len(ratings[ratings['rating']==0.5])
```

Out[42]:
```
239125
```

In [54]:
```
ratings['rating'].mean()
```

Out[54]:
```
3.5255285642993797
```

In [50]:
```
ratings['rating'].mode()
```

Out[50]:
```
0    4.0
Name: rating, dtype: float64
```

In [68]:
```
ratings['rating'].max()
```

Out[68]:
```
5.0
```

In [58]:
```
ratings.corr()
```

Out[58]:

	userId	movieId	rating
userId	1.000000	-0.000850	0.001175
movieId	-0.000850	1.000000	0.002606
rating	0.001175	0.002606	1.000000

In [84]:
```
filter1=ratings[ratings['rating']>10]
filter2=ratings['rating']>4.5
print(filter2)
print(filter1)
```

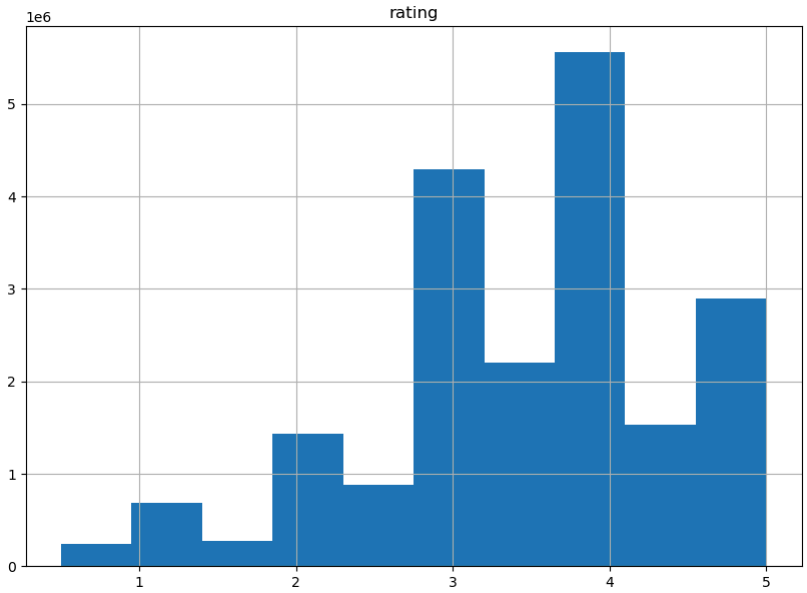
```
0           False
1           False
2           False
3           False
4           False
            ...
20000258    False
20000259    False
20000260    False
20000261     True
20000262    False
Name: rating, Length: 20000263, dtype: bool
Empty DataFrame
Columns: [userId, movieId, rating]
Index: []
```

In [88]:
```
filter1.any()
ratings.rating.head(10)
```

Out[88]:
```
0    3.5
1    3.5
2    3.5
3    3.5
4    3.5
5    3.5
6    4.0
7    4.0
8    4.0
9    4.0
Name: rating, dtype: float64
```

In [86]:
```
%matplotlib inline
ratings.hist(column='rating',figsize=(10,7))
```

Out[86]:
```
array([[<Axes: title={'center': 'rating'}>]], dtype=object)
```



In [102…
```
movies.columns
movies[['title','genres']]
```

Out[102…

	title	genres
0	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	Jumanji (1995)	Adventure|Children|Fantasy
2	Grumpier Old Men (1995)	Comedy|Romance
3	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	Father of the Bride Part II (1995)	Comedy
...	...	...
27273	Kein Bund für's Leben (2007)	Comedy
27274	Feuer, Eis & Dosenbier (2002)	Comedy
27275	The Pirates (2014)	Adventure
27276	Rentun Ruusu (2001)	(no genres listed)
27277	Innocence (2014)	Adventure|Fantasy|Horror

27278 rows × 2 columns

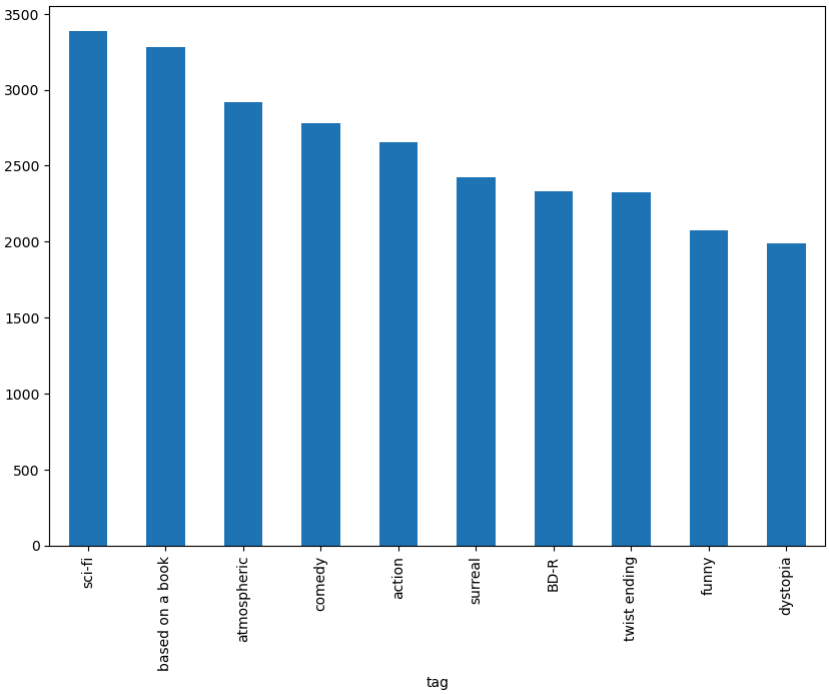
In [124…
```
tag_couns=tags['tag'].value_counts()
print(tag_couns)
tag_couns[:10]
```

```
tag
sci-fi                          3384
based on a book                 3281
atmospheric                     2917
comedy                          2779
action                          2657
                                ...
Paul Adelstein                     1
the wig                            1
killer fish                        1
genetically modified monsters      1
topless scene                      1
Name: count, Length: 38643, dtype: int64
```

Out[124…
```
tag
sci-fi             3384
based on a book    3281
atmospheric        2917
comedy             2779
action             2657
surreal            2427
BD-R               2334
twist ending       2323
funny              2072
dystopia           1991
Name: count, dtype: int64
```

In [126…
```
%matplotlib inline
tag_couns[:10].plot(kind='bar',figsize=(10,7))
```

Out[126…
```
<Axes: xlabel='tag'>
```



In [128…
```
movies['genres'].str.contains('Action')
```

Out[128…
```
0        False
1        False
2        False
3        False
4        False
         ...
27273    False
27274    False
27275    False
27276    False
27277    False
Name: genres, Length: 27278, dtype: bool
```

In [138…
```
is_action=movies['genres'].str.contains('Action')
```

In [140…
```
movies[is_action][5:10]
```

Out[140…

	movieId	title	genres
22	23	Assassins (1995)	Action|Crime|Thriller
41	42	Dead Presidents (1995)	Action|Crime|Drama
43	44	Mortal Kombat (1995)	Action|Adventure|Fantasy
50	51	Guardian Angel (1994)	Action|Drama|Thriller
65	66	Lawnmower Man 2: Beyond Cyberspace (1996)	Action|Sci-Fi|Thriller

In [158…
```
ratings_count=ratings[['movieId','rating']].groupby('rating').count()
```

In [160…
```
print(ratings_count)
```

```
        movieId
rating
0.5      239125
1.0      680732
1.5      279252
2.0     1430997
2.5      883398
3.0     4291193
3.5     2200156
4.0     5561926
4.5     1534824
5.0     2898660
```

In [166…
```
ratings.head(10)
```

Out[166…

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5
2	1	32	3.5
3	1	47	3.5
4	1	50	3.5
5	1	112	3.5
6	1	151	4.0
7	1	223	4.0
8	1	253	4.0
9	1	260	4.0

In [168…
```
avg_ratings=ratings[['movieId','rating']].groupby('movieId').mean()
```

In [170…
```
print(avg_ratings)
```

```
            rating
movieId
1         3.921240
2         3.211977
3         3.151040
4         2.861393
5         3.064592
...            ...
131254    4.000000
131256    4.000000
131258    2.500000
131260    3.000000
131262    4.000000

[26744 rows x 1 columns]
```

In [172…
```
ratings.head(20)
```

Out[172…

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5
2	1	32	3.5
3	1	47	3.5
4	1	50	3.5
5	1	112	3.5
6	1	151	4.0
7	1	223	4.0
8	1	253	4.0
9	1	260	4.0
10	1	293	4.0
11	1	296	4.0
12	1	318	4.0
13	1	337	3.5
14	1	367	3.5
15	1	541	4.0
16	1	589	3.5
17	1	593	3.5
18	1	653	3.0
19	1	919	3.5

In [188…
```
t=movies.merge(tags,on='movieId',how='outer')
```

In [192…
```
t.head(5)
```

Out[192…

	movieId	title	genres	userId	tag
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1644.0	Watched
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741.0	computer animation
2	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741.0	Disney animated feature
3	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741.0	Pixar animation
4	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741.0	TÃ©a Leoni does not star in this movie

In [194…
```
t1=movies.merge(tags,on='movieId',how='inner')
```

In [196…
```
t1.head(5)
```

Out[196…

	movieId	title	genres	userId	tag
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1644	Watched
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	computer animation
2	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	Disney animated feature
3	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	Pixar animation
4	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	TÃ©a Leoni does not star in this movie

In [10]:
```
avg_ratings=ratings.groupby('movieId').mean()
```

In [17]:
```
del avg_ratings['userId']
avg_ratings
```

Out[17]:

movieId	rating
1	3.921240
2	3.211977
3	3.151040
4	2.861393
5	3.064592
…	…
131254	4.000000
131256	4.000000
131258	2.500000
131260	3.000000
131262	4.000000

26744 rows × 1 columns

In [29]:
```
len(avg_ratings)
```

Out[29]: 26744

In [21]:
```
movies
box_office=movies.merge(avg_ratings,on='movieId',how='inner')
```

In [23]:
```
box_office
```

Out[23]:

	movieId	title	genres	rating
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	3.921240
1	2	Jumanji (1995)	Adventure|Children|Fantasy	3.211977
2	3	Grumpier Old Men (1995)	Comedy|Romance	3.151040
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	2.861393
4	5	Father of the Bride Part II (1995)	Comedy	3.064592
…	…	…	…	…
26739	131254	Kein Bund für's Leben (2007)	Comedy	4.000000
26740	131256	Feuer, Eis & Dosenbier (2002)	Comedy	4.000000
26741	131258	The Pirates (2014)	Adventure	2.500000
26742	131260	Rentun Ruusu (2001)	(no genres listed)	3.000000
26743	131262	Innocence (2014)	Adventure|Fantasy|Horror	4.000000

26744 rows × 4 columns

In [31]:
```
avg_ratings1=ratings.groupby('movieId')
```

In [37]:
```
#avg_ratings1.mean()
box_office
```

Out[37]:

	movieId	title	genres	rating
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	3.921240
1	2	Jumanji (1995)	Adventure|Children|Fantasy	3.211977
2	3	Grumpier Old Men (1995)	Comedy|Romance	3.151040
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	2.861393
4	5	Father of the Bride Part II (1995)	Comedy	3.064592
…	…	…	…	…
26739	131254	Kein Bund für's Leben (2007)	Comedy	4.000000
26740	131256	Feuer, Eis & Dosenbier (2002)	Comedy	4.000000
26741	131258	The Pirates (2014)	Adventure	2.500000
26742	131260	Rentun Ruusu (2001)	(no genres listed)	3.000000
26743	131262	Innocence (2014)	Adventure|Fantasy|Horror	4.000000

26744 rows × 4 columns

In [39]:
```
is_adven=box_office['genres'].str.contains('Adventure')
```

In [47]:
```
box_office[is_adven]
```

Out[47]:

	movieId	title	genres	rating
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	3.921240
1	2	Jumanji (1995)	Adventure|Children|Fantasy	3.211977
7	8	Tom and Huck (1995)	Adventure|Children	3.142049
9	10	GoldenEye (1995)	Action|Adventure|Thriller	3.430029
12	13	Balto (1995)	Adventure|Animation|Children	3.272416
...	...	...	...	...
26683	131084	Hui Buh: The Castle Ghost (2006)	Adventure|Comedy|Fantasy	2.500000
26687	131092	Mickey, Donald, Goofy: The Three Musketeers (2...	Adventure|Animation|Children|Comedy	3.000000
26736	131248	Brother Bear 2 (2006)	Adventure|Animation|Children|Comedy|Fantasy	4.000000
26741	131258	The Pirates (2014)	Adventure	2.500000
26743	131262	Innocence (2014)	Adventure|Fantasy|Horror	4.000000

2287 rows × 4 columns

In [49]:
```
is_highly_rated=box_office['rating']>=4.0
```

In [55]:
```
#is_highly_rated
box_office[is_highly_rated]
```

Out[55]:

	movieId	title	genres	rating
27	28	Persuasion (1995)	Drama|Romance	4.057546
46	47	Seven (a.k.a. Se7en) (1995)	Mystery|Thriller	4.053493
49	50	Usual Suspects, The (1995)	Crime|Mystery|Thriller	4.334372
81	82	Antonia's Line (Antonia) (1995)	Comedy|Drama	4.004925
108	110	Braveheart (1995)	Action|Drama|War	4.042534
...	...	...	...	...
26737	131250	No More School (2000)	Comedy	4.000000
26738	131252	Forklift Driver Klaus: The First Day on the Jo...	Comedy|Horror	4.000000
26739	131254	Kein Bund für's Leben (2007)	Comedy	4.000000
26740	131256	Feuer, Eis & Dosenbier (2002)	Comedy	4.000000
26743	131262	Innocence (2014)	Adventure|Fantasy|Horror	4.000000

1757 rows × 4 columns

In [57]:
```
box_office[is_adven & is_highly_rated]
```

Out[57]:

	movieId	title	genres	rating
257	260	Star Wars: Episode IV - A New Hope (1977)	Action|Adventure|Sci-Fi	4.190672
593	599	Wild Bunch, The (1969)	Adventure|Western	4.004726
708	720	Wallace & Gromit: The Best of Aardman Animatio...	Adventure|Animation|Comedy	4.109473
891	908	North by Northwest (1959)	Action|Adventure|Mystery|Romance|Thriller	4.233538
952	969	African Queen, The (1951)	Adventure|Comedy|Romance|War	4.101558
...	...	...	...	...
26611	130586	Itinerary of a Spoiled Child (1988)	Adventure|Drama	4.500000
26655	130996	The Beautiful Story (1992)	Adventure|Drama|Fantasy	5.000000
26667	131050	Stargate SG-1 Children of the Gods - Final Cut...	Adventure|Sci-Fi|Thriller	5.000000
26736	131248	Brother Bear 2 (2006)	Adventure|Animation|Children|Comedy|Fantasy	4.000000
26743	131262	Innocence (2014)	Adventure|Fantasy|Horror	4.000000

113 rows × 4 columns

In [59]:
```
movie_genres=movies['genres'].str.split('|')
```

In [63]:
```
movie_genres1=movies['genres'].str.split('|',expand=True)
```

In [65]:
```
movie_genres1
```

Out[65]:

	0	1	2	3	4	5	6	7	8	9
0	Adventure	Animation	Children	Comedy	Fantasy	None	None	None	None	None
1	Adventure	Children	Fantasy	None	None	None	None	None	None	None
2	Comedy	Romance	None	None	None	None	None	None	None	None
3	Comedy	Drama	Romance	None	None	None	None	None	None	None
4	Comedy	None	None	None	None	None	None	None	None	None
...	...	...	...	...	...	...	...	...	...	...
27273	Comedy	None	None	None	None	None	None	None	None	None
27274	Comedy	None	None	None	None	None	None	None	None	None
27275	Adventure	None	None	None	None	None	None	None	None	None
27276	(no genres listed)	None	None	None	None	None	None	None	None	None
27277	Adventure	Fantasy	Horror	None	None	None	None	None	None	None

27278 rows × 10 columns

In [67]:
```
movies.tail(4)
```

Out[67]:

	movieId	title	genres
27274	131256	Feuer, Eis & Dosenbier (2002)	Comedy
27275	131258	The Pirates (2014)	Adventure
27276	131260	Rentun Ruusu (2001)	(no genres listed)
27277	131262	Innocence (2014)	Adventure|Fantasy|Horror

In [69]:
```
movie_genres.tail(4)
```

Out[69]:
```
27274                      [Comedy]
27275                   [Adventure]
27276           [(no genres listed)]
27277    [Adventure, Fantasy, Horror]
Name: genres, dtype: object
```

In [71]:
```
del movie_genres
```

In [73]:
```
movie_genres1['isComedy']=movies['genres'].str.contains('Comdey')
```

In [75]:
```
movie_genres1
```

Out[75]:

	0	1	2	3	4	5	6	7	8	9	isComedy
0	Adventure	Animation	Children	Comedy	Fantasy	None	None	None	None	None	False
1	Adventure	Children	Fantasy	None	None	None	None	None	None	None	False
2	Comedy	Romance	None	None	None	None	None	None	None	None	False
3	Comedy	Drama	Romance	None	None	None	None	None	None	None	False
4	Comedy	None	None	None	None	None	None	None	None	None	False
...	...	...	...	...	...	...	...	...	...	...	...
27273	Comedy	None	None	None	None	None	None	None	None	None	False
27274	Comedy	None	None	None	None	None	None	None	None	None	False
27275	Adventure	None	None	None	None	None	None	None	None	None	False
27276	(no genres listed)	None	None	None	None	None	None	None	None	None	False
27277	Adventure	Fantasy	Horror	None	None	None	None	None	None	None	False

27278 rows × 11 columns

In [77]:
```
movies.head(4)
```

Out[77]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance

In [99]:
```
movies['Year']=movies['title'].str.extract(r'* \((.*) \).*',expand=True)
#str1='tail(123)'
#type(str1)
```

```
---------------------------------------------------------------------------
error                                     Traceback (most recent call last)
Cell In[99], line 1
----> 1 movies['Year']=movies['title'].str.extract(r'* \((.*) \).*',expand=True)

File ~\anaconda3\Lib\site-packages\pandas\core\strings\accessor.py:137, in forbid_nonstring_types.<locals>._forbid_nonstring_types.<locals>.wrapper(self, *args, **kwargs)
    132     msg = (
    133         f"Cannot use .str.{func_name} with values of "
    134         f"inferred dtype '{self._inferred_dtype}'."
    135     )
    136     raise TypeError(msg)
--> 137 return func(self, *args, **kwargs)

File ~\anaconda3\Lib\site-packages\pandas\core\strings\accessor.py:2738, in StringMethods.extract(self, pat, flags, expand)
   2735 if not isinstance(expand, bool):
   2736     raise ValueError("expand must be True or False")
-> 2738 regex = re.compile(pat, flags=flags)
   2739 if regex.groups == 0:
   2740     raise ValueError("pattern contains no capture groups")

File ~\anaconda3\Lib\re\__init__.py:228, in compile(pattern, flags)
    226 def compile(pattern, flags=0):
    227     "Compile a regular expression pattern, returning a Pattern object."
--> 228     return _compile(pattern, flags)

File ~\anaconda3\Lib\re\__init__.py:307, in _compile(pattern, flags)
    301     import warnings
    302     warnings.warn("The re.TEMPLATE/re.T flag is deprecated "
    303             "as it is an undocumented flag "
    304             "without an obvious purpose. "
    305             "Don't use it.",
    306             DeprecationWarning)
--> 307 p = _compiler.compile(pattern, flags)
    308 if flags & DEBUG:
    309     return p

File ~\anaconda3\Lib\re\_compiler.py:745, in compile(p, flags)
    743 if isstring(p):
    744     pattern = p
--> 745     p = _parser.parse(p, flags)
    746 else:
    747     pattern = None

File ~\anaconda3\Lib\re\_parser.py:979, in parse(str, flags, state)
    976 state.flags = flags
    977 state.str = str
--> 979 p = _parse_sub(source, state, flags & SRE_FLAG_VERBOSE, 0)
    980 p.state.flags = fix_flags(str, p.state.flags)
    982 if source.next is not None:

File ~\anaconda3\Lib\re\_parser.py:460, in _parse_sub(source, state, verbose, nested)
    458 start = source.tell()
    459 while True:
--> 460     itemsappend(_parse(source, state, verbose, nested + 1,
    461                        not nested and not items))
    462     if not sourcematch("|"):
    463         break

File ~\anaconda3\Lib\re\_parser.py:687, in _parse(source, state, verbose, nested, first)
    685     item = None
    686 if not item or item[0][0] is AT:
--> 687     raise source.error("nothing to repeat",
    688                        source.tell() - here + len(this))
    689 if item[0][0] in _REPEATCODES:
    690     raise source.error("multiple repeat",
    691                        source.tell() - here + len(this))

error: nothing to repeat at position 0
```

In [93]:
```
str1.split('(')
```

Out[93]:
```
['tail', '123)']
```

In [104…
```
tags.head(4)
```

Out[104…

	userId	movieId	tag	timestamp
0	18	4141	Mark Waters	2009-04-24 18:19:40
1	65	208	dark hero	2013-05-10 01:41:18
2	65	353	dark hero	2013-05-10 01:41:19
3	65	521	noir thriller	2013-05-10 01:39:43

In [109…
```
tags['parsed_time']=pd.to_datetime(tags['timestamp'])
```

In [111…
```
tags
```

Out[111…

	userId	movieId	tag	timestamp	parsed_time
0	18	4141	Mark Waters	2009-04-24 18:19:40	2009-04-24 18:19:40
1	65	208	dark hero	2013-05-10 01:41:18	2013-05-10 01:41:18
2	65	353	dark hero	2013-05-10 01:41:19	2013-05-10 01:41:19
3	65	521	noir thriller	2013-05-10 01:39:43	2013-05-10 01:39:43
4	65	592	dark hero	2013-05-10 01:41:18	2013-05-10 01:41:18
...	...	...	...	...	...
465559	138446	55999	dragged	2013-01-23 23:29:32	2013-01-23 23:29:32
465560	138446	55999	Jason Bateman	2013-01-23 23:29:38	2013-01-23 23:29:38
465561	138446	55999	quirky	2013-01-23 23:29:38	2013-01-23 23:29:38
465562	138446	55999	sad	2013-01-23 23:29:32	2013-01-23 23:29:32
465563	138472	923	rise to power	2007-11-02 21:12:47	2007-11-02 21:12:47

465564 rows × 5 columns

In [ ]:
```

```