# 11-11-2025 KAGGLE

# Pandas With Data Science.AI

# MovieLens 20M Dataset

https://www.kaggle.com/code/harunshimanto/pandas-with-data-science-ai

In [1]:
```
import pandas as pd # import libraries
```

In [2]:
```
ratings = pd.read_csv(r"C:\Users\karthik reddy\Downloads\archive\rating.csv")
```

In [3]:
```
ratings.head()
```

Out[3]:

| | 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 [4]:
```
ratings.head(3)
```

Out[4]:

| | 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 |

In [5]:
```
ratings.shape
```

Out[5]:
```
(20000263, 4)
```

In [6]:
```
tags = pd.read_csv(r"C:\Users\karthik reddy\Downloads\archive\tag.csv")
tags.shape
```

Out[6]:
```
(465564, 4)
```

In [7]:
```
tags.head()
```

Out[7]:

| | 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 [8]:
```
tags.head(3)
```

Out[8]:

| | 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 |

In [9]:
```
movies = pd.read_csv(r"C:\Users\karthik reddy\Downloads\archive\movie.csv")
movies.shape
```

Out[9]:
```
(27278, 3)
```

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

Out[10]:

| | 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 |

In [11]:
```
movies.head(3)
```

Out[11]:

| | 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 |

In [12]:
```
print(tags.columns)
print(ratings.columns)
print(movies.columns)
```

```
Index(['userId', 'movieId', 'tag', 'timestamp'], dtype='object')
Index(['userId', 'movieId', 'rating', 'timestamp'], dtype='object')
Index(['movieId', 'title', 'genres'], dtype='object')
```

In [14]:
```
del ratings['timestamp']
del tags['timestamp']
```

In [15]:
```
print(tags.columns)
print(ratings.columns)
print(movies.columns)
```

```
Index(['userId', 'movieId', 'tag'], dtype='object')
Index(['userId', 'movieId', 'rating'], dtype='object')
Index(['movieId', 'title', 'genres'], dtype='object')
```

In [16]:
```
tags.head()
```

Out[16]:

| | userId | movieId | tag |
|---|---|---|---|
| **0** | 18 | 4141 | Mark Waters |
| **1** | 65 | 208 | dark hero |
| **2** | 65 | 353 | dark hero |
| **3** | 65 | 521 | noir thriller |
| **4** | 65 | 592 | dark hero |

In [17]:
```
tags.head(3)
```

Out[17]:

| | userId | movieId | tag |
|---|---|---|---|
| **0** | 18 | 4141 | Mark Waters |
| **1** | 65 | 208 | dark hero |
| **2** | 65 | 353 | dark hero |

In [24]:
```
#series

tags.iloc[0]
```

Out[24]:
```
userId              18
movieId           4141
tag        Mark Waters
Name: 0, dtype: object
```

In [25]:
```
tags.iloc[2]
```

Out[25]:
```
userId            65
movieId          353
tag        dark hero
Name: 2, dtype: object
```

In [26]:
```
tags.iloc[0]
```

Out[26]:
```
userId              18
movieId           4141
tag        Mark Waters
Name: 0, dtype: object
```

In [27]:
```
row_0 = tags.iloc[0]
row_0
```

Out[27]:
```
userId              18
movieId           4141
tag        Mark Waters
Name: 0, dtype: object
```

In [28]:
```
row_1 = tags.iloc[1]
row_1
```

Out[28]:
```
userId            65
movieId          208
tag        dark hero
Name: 1, dtype: object
```

In [29]:
```
row_0.index
```

Out[29]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [30]:
```
row_0 = tags.iloc[0]
type(row_0)
```

Out[30]:
```
pandas.core.series.Series
```

In [31]:
```
print(row_0)
```

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

In [32]:
```
row_0.index
```

Out[32]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [33]:
```
row_0['userId']
```

Out[33]:
```
np.int64(18)
```

In [34]:
```
'rating' in row_0
```

Out[34]:
```
False
```

In [35]:
```
row_0.name
```

Out[35]:
```
0
```

In [36]:
```
row_0 = row_0.rename('firstRow')
row_0.name
```

Out[36]: 'firstRow'

# Data Frames

In [37]:
```
tags.head()
```

Out[37]:

| | userId | movieId | tag |
|---|---|---|---|
| **0** | 18 | 4141 | Mark Waters |
| **1** | 65 | 208 | dark hero |
| **2** | 65 | 353 | dark hero |
| **3** | 65 | 521 | noir thriller |
| **4** | 65 | 592 | dark hero |

In [38]:
```
tags.index
```

Out[38]:
```
RangeIndex(start=0, stop=465564, step=1)
```

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

Out[39]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [40]:
```
tags.iloc[ [0,11,500] ]
```

Out[40]:

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

Descriptive Statistics

In [41]:
```
ratings['rating'].describe()
```

Out[41]:
```
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 [42]:
```
ratings.describe()
```

Out[42]:

| | userId | movieId | rating |
|---|---|---|---|
| **count** | 2.000026e+07 | 2.000026e+07 | 2.000026e+07 |
| **mean** | 6.904587e+04 | 9.041567e+03 | 3.525529e+00 |
| **std** | 4.003863e+04 | 1.978948e+04 | 1.051989e+00 |
| **min** | 1.000000e+00 | 1.000000e+00 | 5.000000e-01 |
| **25%** | 3.439500e+04 | 9.020000e+02 | 3.000000e+00 |
| **50%** | 6.914100e+04 | 2.167000e+03 | 3.500000e+00 |
| **75%** | 1.036370e+05 | 4.770000e+03 | 4.000000e+00 |
| **max** | 1.384930e+05 | 1.312620e+05 | 5.000000e+00 |

In [43]:
```
ratings['rating'].mean()
```

Out[43]:
```
np.float64(3.5255285642993797)
```

In [44]:
```
ratings.mean()
```

Out[44]:
```
userId     69045.872583
movieId     9041.567330
rating         3.525529
dtype: float64
```

In [45]:
```
ratings['rating'].min()
```

Out[45]:
```
0.5
```

In [46]:
```
ratings['rating'].max()
```

Out[46]:
```
5.0
```

In [47]:
```
ratings['rating'].std()
```

Out[47]:
```
1.051988919275684
```

In [48]:
```
ratings['rating'].mode()
```

Out[48]:
```
0    4.0
Name: rating, dtype: float64
```

In [49]:
```
ratings.corr()
```

Out[49]:

| | 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 [50]:
```
filter1 = ratings['rating'] > 10
print(filter1)
filter1.any()
```

```
0           False
1           False
2           False
3           False
4           False
            ...
20000258    False
20000259    False
20000260    False
20000261    False
20000262    False
Name: rating, Length: 20000263, dtype: bool
```

Out[50]: np.False_

In [51]:
```
filter2 = ratings['rating'] > 0
filter2.all()
```

Out[51]: np.True_

## Data Cleaning: Handling Missing Data

In [52]:
```
movies.shape
```

Out[52]: (27278, 3)

In [53]:
```
movies.isnull().any().any()
```

Out[53]: np.False_

In [54]:
```
ratings.shape
```

Out[54]: (20000263, 3)

In [55]:
```
ratings.isnull().any().any()
```

Out[55]: np.False_

In [56]:
```
tags.shape
```

Out[56]: (465564, 3)

In [57]:
```
tags.isnull().any().any()
```

Out[57]: np.True_

In [58]:
```
tags=tags.dropna()
```

In [59]:
```
tags.isnull().any().any()
```

Out[59]:
```
np.False_
```

In [60]:
```
tags.shape
```

Out[60]:
```
(465548, 3)
```

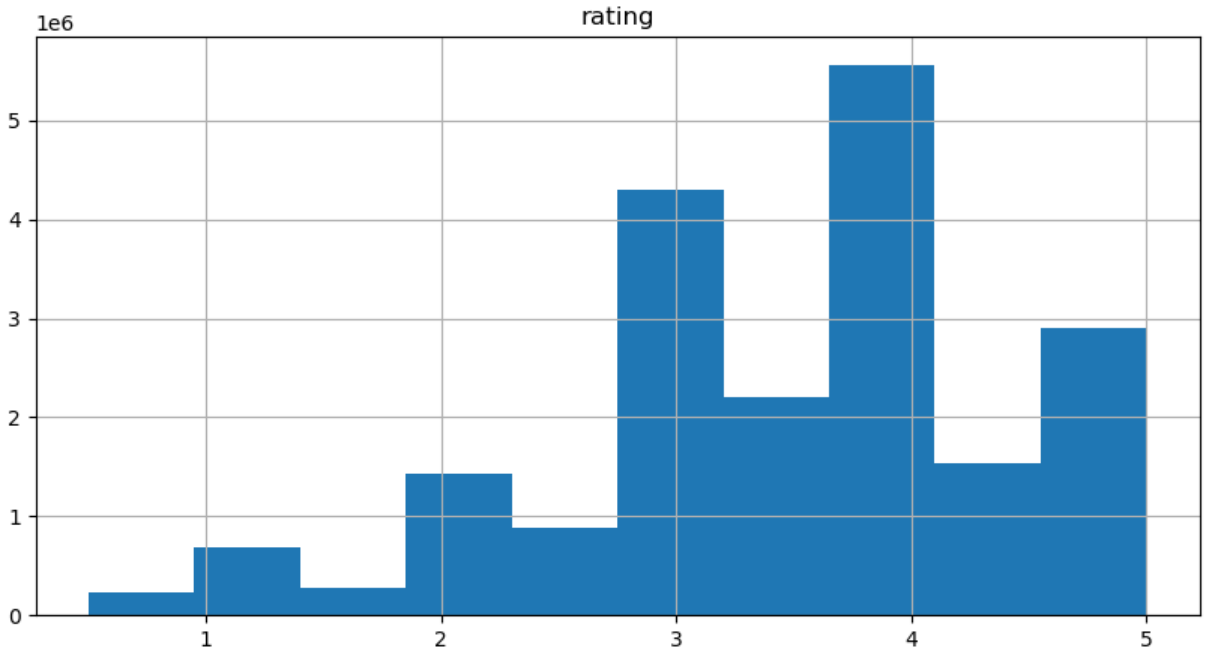
Data Visualization

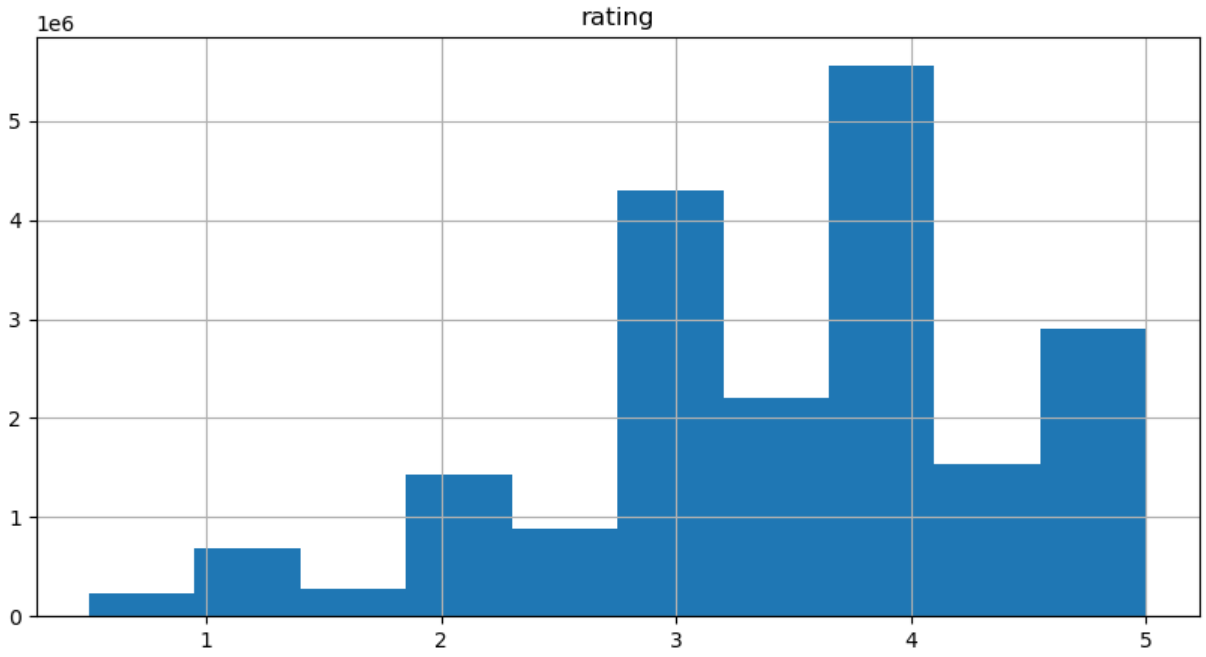
In [66]:
```
%matplotlib inline
ratings.hist(column='rating', figsize=(10,5))
```

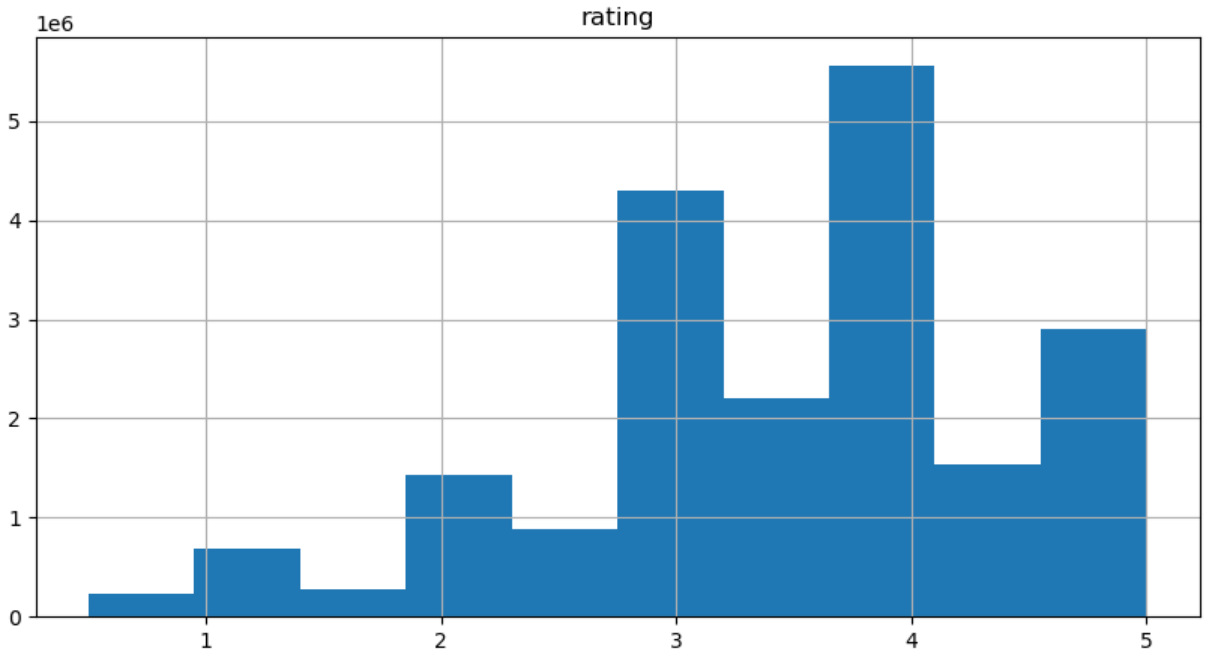
Out[66]:
```
array([[<Axes: title={'center': 'rating'}>]], dtype=object)
```

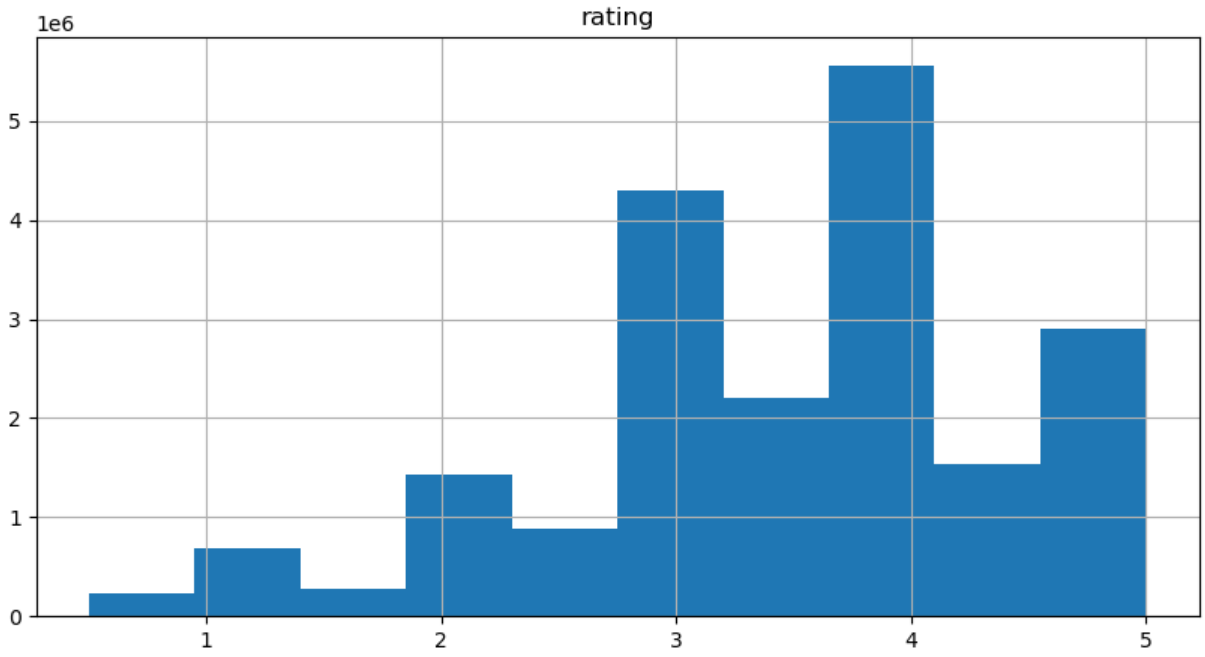
In [67]:
```
import matplotlib.pyplot as plt

%matplotlib inline
ratings.hist(column='rating', figsize=(10,5))
plt.show()
```




rating
1e6
0
1
2
3
4
5
1
2
3
4
5


rating
1e6
0
1
2
3
4
5
1
2
3
4
5

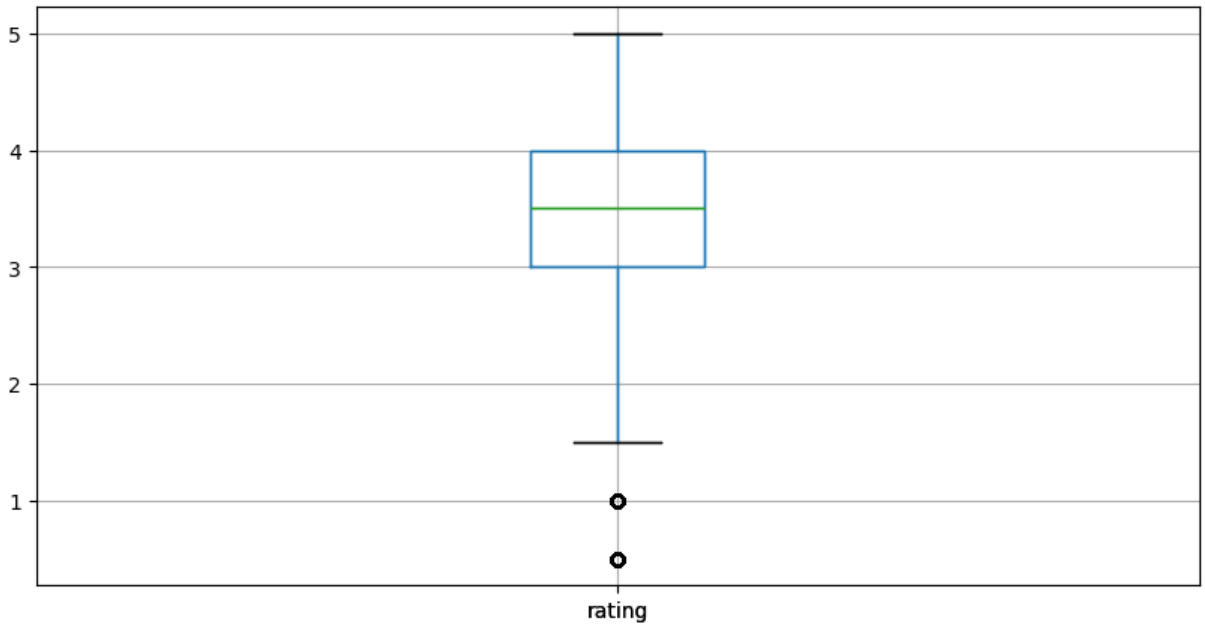



In [68]:
```
ratings.boxplot(column='rating', figsize=(10,5))
```

Out[68]: <Axes: >

In [69]:
```
%matplotlib inline
import matplotlib.pyplot as plt

ratings.boxplot(column='rating', figsize=(10,5))
plt.show()
```




## Slicing Out Columns

In [70]:
```
tags['tag'].head()
```

Out[70]:
```
0      Mark Waters
1        dark hero
2        dark hero
3    noir thriller
4        dark hero
Name: tag, dtype: object
```

In [71]:
```
movies[['title','genres']].head()
```

Out[71]:

| | 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 |

In [72]:
```
ratings[-10:]
```

Out[72]:

| | userId | movieId | rating |
|---|---|---|---|
| **20000253** | 138493 | 60816 | 4.5 |
| **20000254** | 138493 | 61160 | 4.0 |
| **20000255** | 138493 | 65682 | 4.5 |
| **20000256** | 138493 | 66762 | 4.5 |
| **20000257** | 138493 | 68319 | 4.5 |
| **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 |

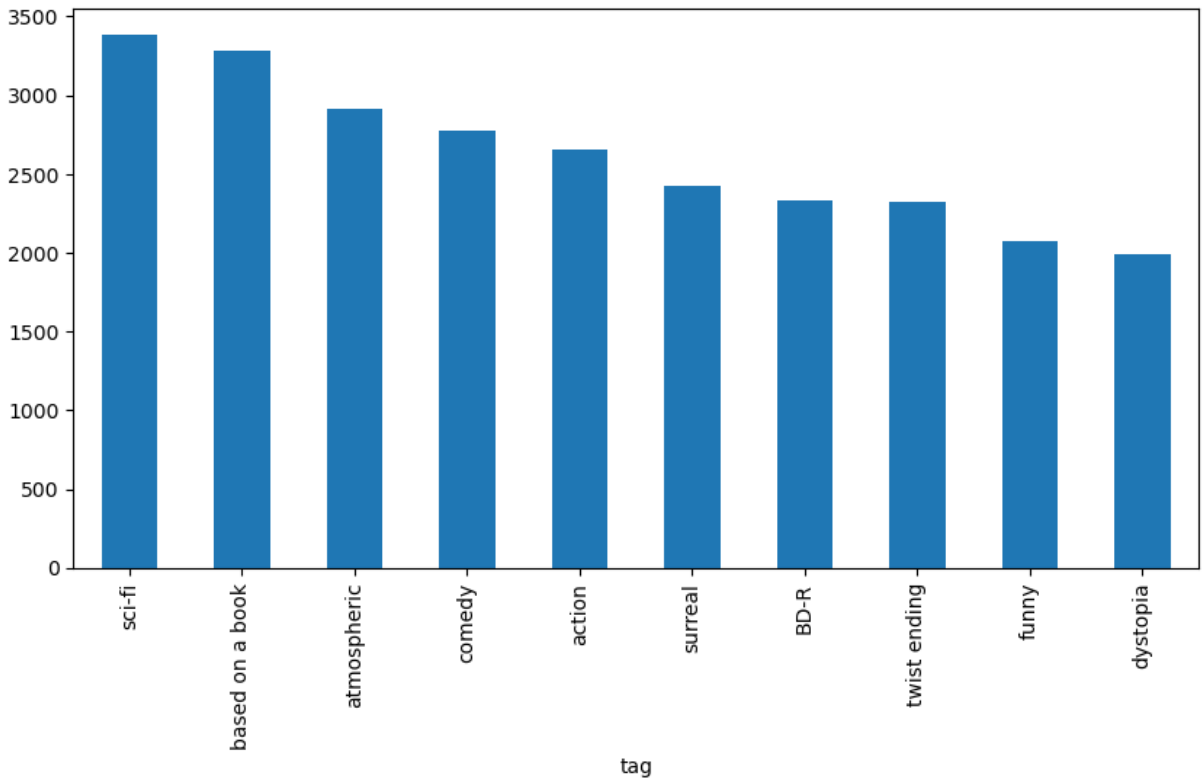
In [73]:
```
tag_counts = tags['tag'].value_counts()
tag_counts[-10:]
```

Out[73]:
```
tag
chiptunes          1
ewan macgregor     1
Disguises          1
retarted           1
operatic           1
heartrending       1
film crew          1
es                 1
girltalk           1
Spanish films      1
Name: count, dtype: int64
```

In [74]:
```
tag_counts[:10].plot(kind='bar', figsize=(10,5))
```

Out[74]:
```
<Axes: xlabel='tag'>
```

In [75]:
```
plt.show()
```




In [ ]: