Day 19 – MovieLens Data Analysis with Pandas

In this notebook, I worked with the **MovieLens 20M dataset** — a popular real-world dataset for building and analyzing movie recommendation systems

Dataset Files Used:

- `movies.csv` → Contains movie metadata like title and genres
- `ratings.csv` → Contains user ratings for movies
- `tags.csv` → Contains user-generated tags associated with movies

All datasets were loaded using **Pandas**, and basic exploratory operations such as shape, column extraction, and cleaning were performed.

Objectives:

- Load and inspect multiple CSV files
- Clean and prepare the data
- Understand relationships between users, movies, and ratings

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

In [2]:
```
movies = pd.read_csv(r"C:\Users\Arman\Downloads\dataset\archive\movie.csv")
```

In [3]:
```
print(type(movies))
```

```
<class 'pandas.core.frame.DataFrame'>
```

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

Out[4]:

	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 [5]:
```
movies.shape
```

Out[5]:
```
(27278, 3)
```

In [6]:
```
ratings = pd.read_csv(r"C:\Users\Arman\Downloads\dataset\archive\rating.csv")
```

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

Out[7]:

	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 [8]:
```
ratings.shape
```
Out[8]:
```
(20000263, 4)
```

In [9]:
```
tags = pd.read_csv(r"C:\Users\Arman\Downloads\dataset\archive\tag.csv")
```

In [10]:
```
tags.head()
```
Out[10]:

	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 [11]:
```
tags.shape
```
Out[11]:
```
(465564, 4)
```

In [12]:
```
tags.columns
```
Out[12]:
```
Index(['userId', 'movieId', 'tag', 'timestamp'], dtype='object')
```

In [13]:
```
ratings.columns
```
Out[13]:
```
Index(['userId', 'movieId', 'rating', 'timestamp'], dtype='object')
```

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

In [15]:
```
ratings
```

Out[15]:

	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 × 3 columns

In [16]:
```
tags
```

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
...	...	...	...
465559	138446	55999	dragged
465560	138446	55999	Jason Bateman
465561	138446	55999	quirky
465562	138446	55999	sad
465563	138472	923	rise to power

465564 rows × 3 columns

Data Structure

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

Out[17]:

	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 [18]:
```
tags.iloc[2]
```
Out[18]:
```
userId             65
movieId           353
tag         dark hero
Name: 2, dtype: object
```
In [19]:
```
row_0 = tags.iloc[0]
row_0
```
Out[19]:
```
userId                18
movieId             4141
tag          Mark Waters
Name: 0, dtype: object
```
In [20]:
```
type(row_0)
```
Out[20]:
```
pandas.core.series.Series
```
In [21]:
```
row_0.index
```
Out[21]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```
In [22]:
```
print(row_0['userId'])
```
```
18
```
In [23]:
```
'rating' in row_0
```
Out[23]:
```
False
```
In [24]:
```
row_0.name
```
Out[24]:
```
0
```
In [25]:
```
row_0 = row_0.rename('firstRow')
row_0
```
Out[25]:
```
userId                18
movieId             4141
tag          Mark Waters
Name: firstRow, dtype: object
```
In [26]:
```
row_0.name
```
Out[26]:
```
'firstRow'
```

DataFrames

In [27]:
```
tags.head()
```
Out[27]:

	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 [28]:
```
tags.index
```
Out[28]:
```
RangeIndex(start=0, stop=465564, step=1)
```

```
In [29]: tags.columns
```

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

```
In [30]: tags.iloc[[0,11,500]]
```

Out[30]:

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

Descriptive Statistics

```
In [31]: ratings
```

Out[31]:

	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 × 3 columns

```
In [32]: ratings['rating'].describe()
```

```
Out[32]: 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 [33]: ratings.describe()
```

Out[33]:

	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 [34]:
```
ratings['rating'].mean()
```
Out[34]:
```
np.float64(3.5255285642993797)
```

In [35]:
```
ratings.mean()
```
Out[35]:
```
userId      69045.872583
movieId      9041.567330
rating          3.525529
dtype: float64
```

In [36]:
```
ratings['rating'].min()
```
Out[36]:
```
0.5
```

In [37]:
```
ratings['rating'].max()
```
Out[37]:
```
5.0
```

In [38]:
```
ratings['rating'].std()
```
Out[38]:
```
1.051988919275684
```

In [39]:
```
ratings['rating'].mode()
```
Out[39]:
```
0    4.0
Name: rating, dtype: float64
```

In [40]:
```
ratings.corr()
```
Out[40]:

	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 [41]:
```
filter1 = ratings['rating'] > 10
print(filter1)
```

```
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
```

In [42]:
```
filter1.any()
```

Out[42]:
```
np.False_
```

In [43]:
```
filter2 = ratings['rating'] > 0
print(filter2)
```

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

In [44]:
```
print(filter2.all())
```

```
True
```

Data Cleaning: Handling Missing Data

In [45]:
```
movies
```

Out[45]:

	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
...	...	...	...
27273	131254	Kein Bund für's Leben (2007)	Comedy
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

27278 rows × 3 columns

In [46]:
```
movies.shape
```

Out[46]:
```
(27278, 3)
```

```
In [47]: movies.isnull().any().any()
```

```
Out[47]: np.False_
```

```
In [48]: ratings.shape
```

```
Out[48]: (20000263, 3)
```

```
In [49]: ratings.isnull().any().any()
```

```
Out[49]: np.False_
```

```
In [50]: tags.shape
```

```
Out[50]: (465564, 3)
```

```
In [51]: tags.isnull().any().any()
```

```
Out[51]: np.True_
```

```
In [52]: tags = tags.dropna()
```

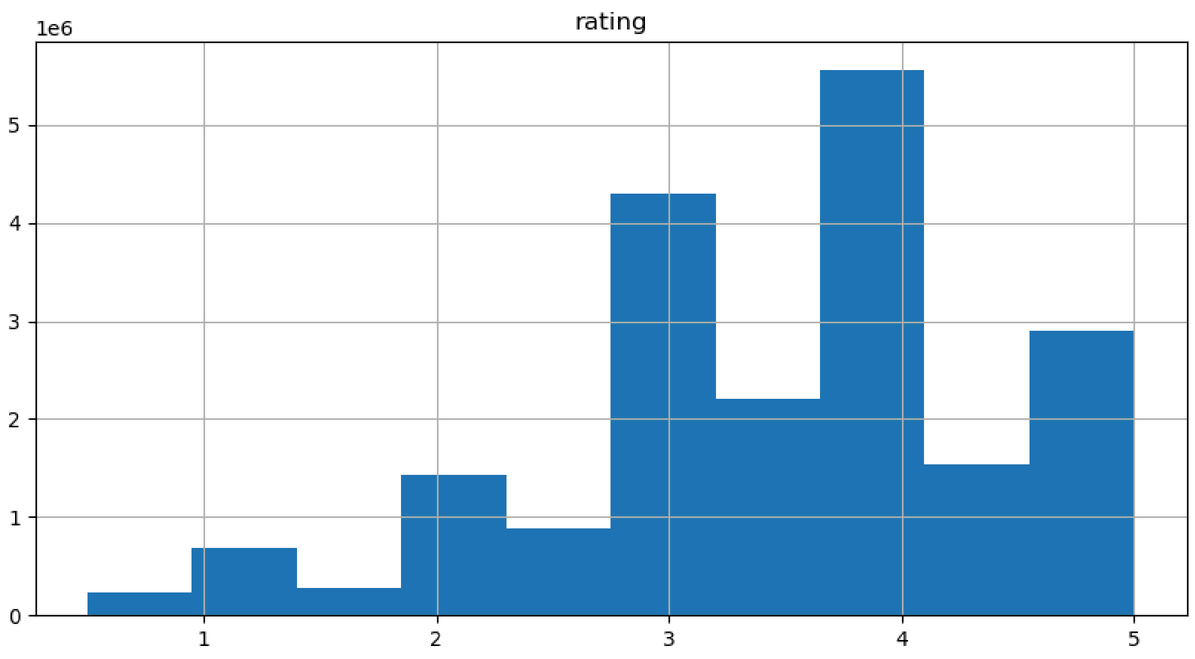
```
In [53]: tags.isnull().any().any()
```

```
Out[53]: np.False_
```

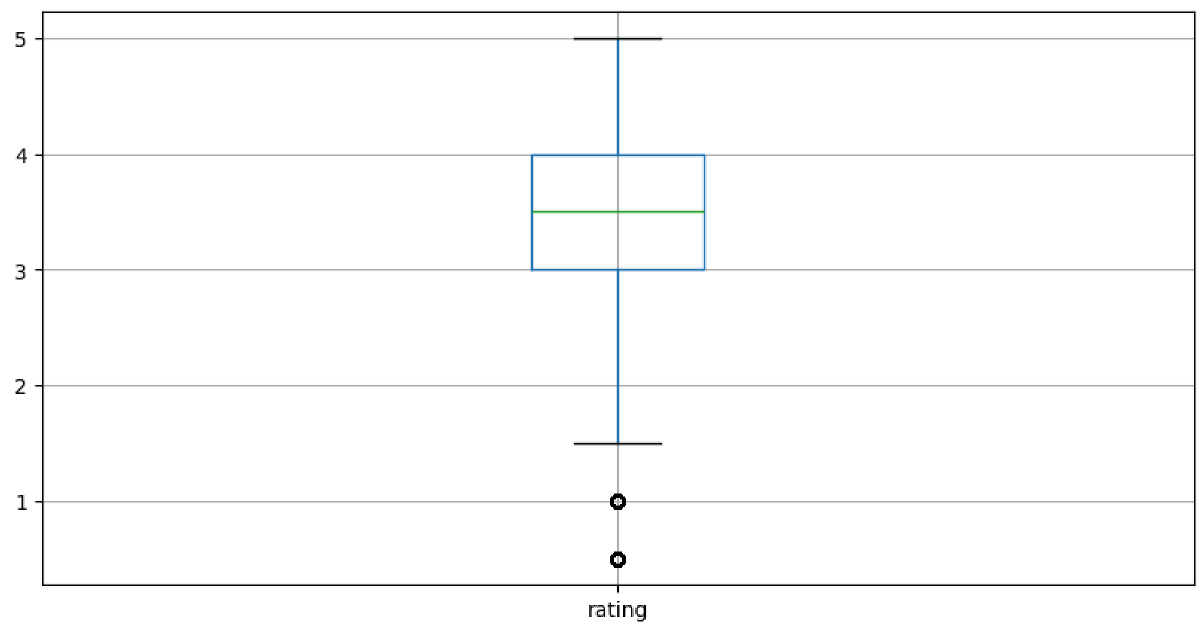
Data Visualization

```
In [54]: import matplotlib.pyplot as plt
         %matplotlib inline
```

```
In [55]: ratings.hist(column='rating', figsize=(10,5))
         plt.show()
```



```
In [56]: ratings.boxplot(column='rating', figsize=(10,5))
         plt.show()
```



Slicing Out Columns

In [57]:
```
tags['tag'].head()
```

Out[57]:
```
0      Mark Waters
1        dark hero
2        dark hero
3    noir thriller
4        dark hero
Name: tag, dtype: object
```

In [58]:
```
movies[['title','genres']].head()
```

Out[58]:

	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 [59]:
```
ratings[-10:]
```

Out[59]:

	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 [60]:

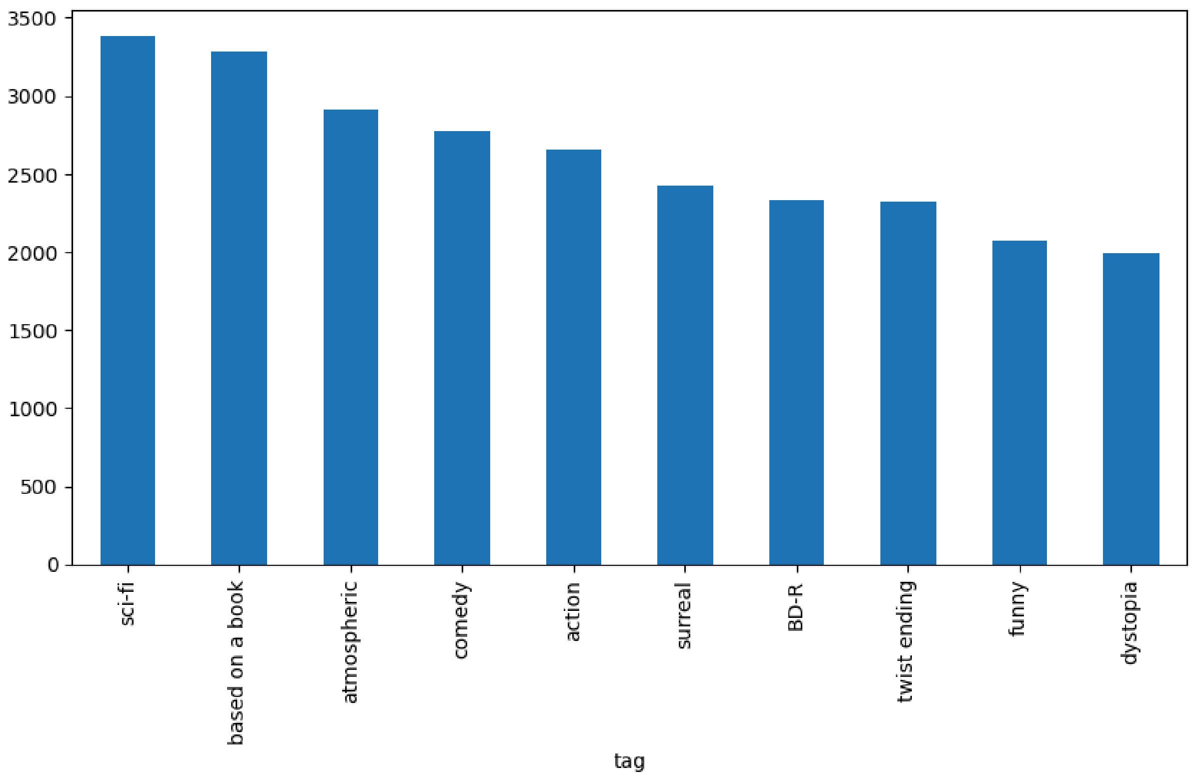
```
tag_counts = tags['tag'].value_counts()
tag_counts[-10:]
```

Out[60]:

```
tag
Hell naw                        1
This is my happy face           1
I heel toe on Uday's house      1
Why?                            1
Bobo                            1
Diamond Dallas Page             1
I'm Devon Butler!               1
No arguement                    1
Really Bad                      1
Botox                           1
Name: count, dtype: int64
```

In [61]:

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


sci-fi
based on a book
atmospheric
comedy
action
surreal
BD-R
twist ending
funny
dystopia
tag