Kaggle task for data science with panda ai

In [2]:
```
import pandas as pd # import library
```

In [56]:
```
rating = pd.read_csv(r'C:\Users\Hp\Downloads\archive\rating.csv')
rating.head(3)
```

Out[56]:

	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 [18]:
```
movie = pd.read_csv(r'C:\Users\Hp\Downloads\archive\movie.csv')
movie.head(3)
```

Out[18]:

	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 [16]:
```
tag = pd.read_csv(r'C:\Users\Hp\Downloads\archive\tag.csv')
tag.head(3)
```

Out[16]:

	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

. for current analysis we'll remove timestamp

In [19]:
```
del rating['timestamp']
del tag['timestamp']
```

In [20]:
```
rating.head(2)
```

Out[20]:

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5

In [21]:
```
tag.head(2)
```

Out[21]:

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero

Data Structure

series

In [23]:
```
row_0 = tag.iloc[0]
type(row_0)
```

Out[23]:
```
pandas.core.series.Series
```

In [24]:
```
row_0
```

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

In [25]:
```
row_0.index # gives colums / attributes name
```

Out[25]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [26]:
```
tag['userId']
```

Out[26]:
```
0             18
1             65
2             65
3             65
4             65
            ...
465559    138446
465560    138446
465561    138446
465562    138446
465563    138472
Name: userId, Length: 465564, dtype: int64
```

In [27]:
```
row_0['userId']
```

Out[27]:
```
np.int64(18)
```

In [28]:
```
'rating' in row_0
```

Out[28]:
```
False
```

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

Out[29]:
```
0
```

In [30]:
```
row_0 = row_0.rename('firstRow')
row_0.name
```

Out[30]:
```
'firstRow'
```

DataFrames

In [32]:
```
tag.head()
```

Out[32]:

	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 [33]:
```
tag.index
```

Out[33]:
```
RangeIndex(start=0, stop=465564, step=1)
```

In [34]:
```
tag.columns
```

Out[34]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [35]:
```
tag.iloc[[0,11,500]]
```

Out[35]:

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

Descriptive Statistics

In [36]:
```
rating['rating'].describe()
```

Out[36]:
```
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 [37]:
```
rating.describe()
```

Out[37]:

	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 [38]:
```
rating['rating'].mean()
```

Out[38]:
```
np.float64(3.5255285642993797)
```

In [39]:
```
rating.mean()
```

Out[39]:
```
userId     69045.872583
movieId     9041.567330
rating         3.525529
dtype: float64
```

In [40]:
```
rating.min()
```

Out[40]:
```
userId     1.0
movieId    1.0
rating     0.5
dtype: float64
```

In [41]:
```
rating['rating'].max()
```

Out[41]:
```
5.0
```

In [49]:
```
rating['rating'].min()
```

Out[49]:
```
0.5
```

In [50]:
```
rating['rating'].std()
```

Out[50]:
```
1.051988919275684
```

In [51]:
```
rating['rating'].mode()
```

Out[51]:
```
0    4.0
Name: rating, dtype: float64
```

In [52]:
```
rating.corr()
```

Out[52]:

	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 [58]:
```
filter1 = rating ['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[58]: np.False_

In [59]:
```
filter2 = rating['rating'] > 0
filter2.all()
```

Out[59]: np.True_

Data cleaning: Handling Missing Data

In [61]:
```
movie.shape
```

Out[61]: (27278, 3)

In [62]:
```
movie.isnull().any().any()
```

Out[62]: np.False_

.thats nice no null values

In [64]:
```
rating.shape
```

Out[64]: (20000263, 4)

In [65]:
```
rating.isnull().any().any()
```

Out[65]: np.False_

In [66]:
```
tag.shape
```

Out[66]: (465564, 3)

In [67]:
```
tag=tag.dropna()
```

In [68]:
```
tag.isnull().any().any()
```

Out[68]:
```
np.False_
```

In [69]:
```
tag.shape
```

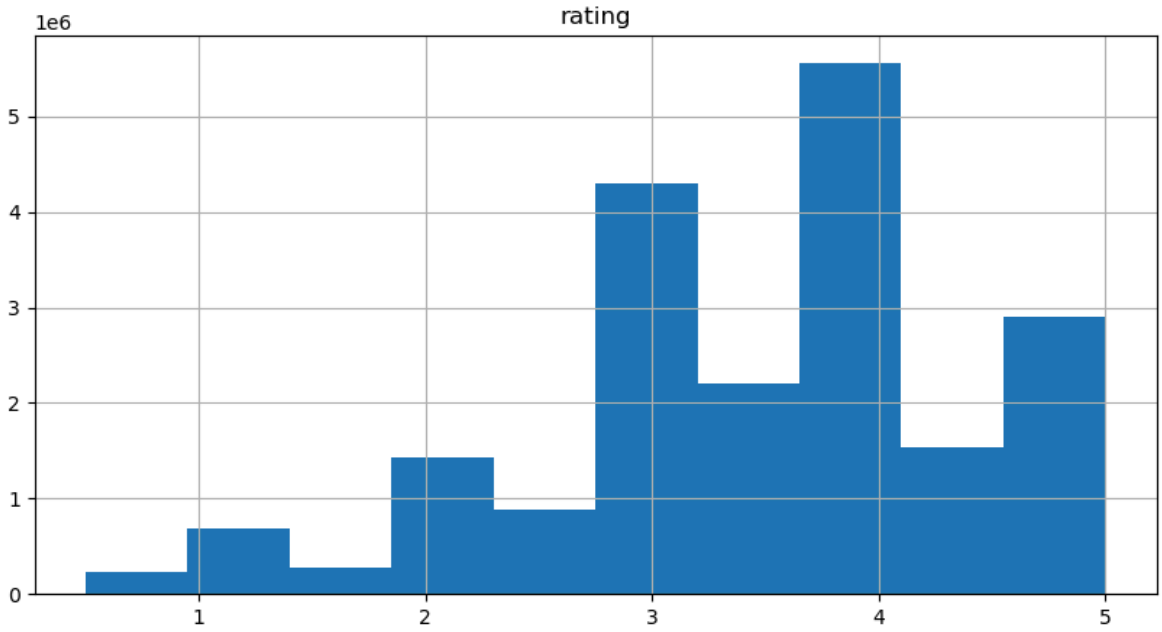
Out[69]:
```
(465548, 3)
```

In [ ]:
```
.Thats nice, No null values! notices the no. of line reduced
```

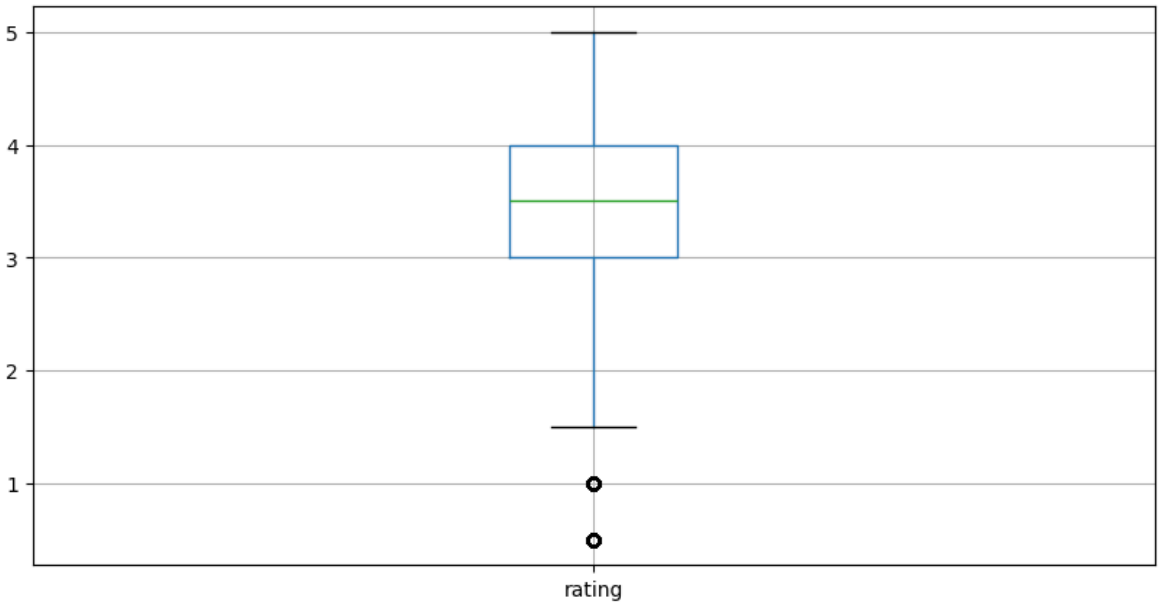
Data visualization

In [72]:
```
import matplotlib.pyplot as plt
%matplotlib inline

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



In [80]:
```
rating.boxplot(column = 'rating', figsize=(10, 5))
plt.show()
```



slicing out columns

In [74]:
```
tag['tag']. head()
```

Out[74]:
```
0      Mark Waters
1        dark hero
2        dark hero
3    noir thriller
4        dark hero
Name: tag, dtype: object
```

In [79]:
```
movie[['title', 'genres']].head()
```

Out[79]:

	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 [82]:
```
rating[-10:]
```

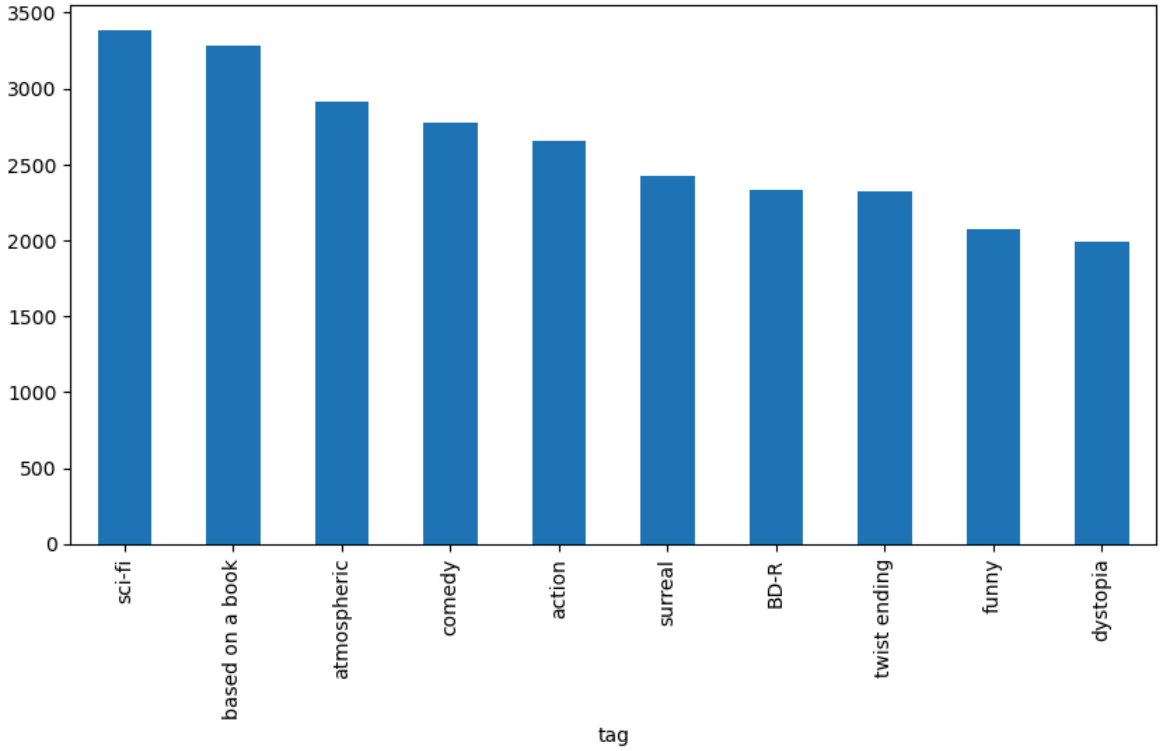
Out[82]:

	userId	movieId	rating	timestamp
20000253	138493	60816	4.5	2009-12-03 18:32:43
20000254	138493	61160	4.0	2009-11-16 16:55:37
20000255	138493	65682	4.5	2009-10-17 21:52:53
20000256	138493	66762	4.5	2009-10-17 18:50:08
20000257	138493	68319	4.5	2009-12-07 18:15:20
20000258	138493	68954	4.5	2009-11-13 15:42:00
20000259	138493	69526	4.5	2009-12-03 18:31:48
20000260	138493	69644	3.0	2009-12-07 18:10:57
20000261	138493	70286	5.0	2009-11-13 15:42:24
20000262	138493	71619	2.5	2009-10-17 20:25:36

In [84]:
```
tag_counts = tag['tag']. value_counts()
tag_counts[-10:]
```

Out[84]:
```
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 [87]:
```
tag_counts[:10].plot(kind = 'bar', figsize=(10,5)),
colour = ['m', 'orange', 'green', 'red', 'brown', 'pink', 'grey', 'yellow', 'tea
plt.show()
```


3500
3000
2500
2000
1500
1000
500
0
sci-fi
based on a book
atmospheric
comedy
action
surreal
BD-R
twist ending
funny
dystopia
tag

In [ ]: