KAGGLE INTRODUCTION

Project 4: IMDB Movie Rating Analysis

In [17]:
```
import pandas as pd
```

In [19]:
```
ratings=pd.read_csv(r"C:\Users\HP\OneDrive\Downloads\archive (1)\rating.csv")
```

In [20]:
```
print(type(ratings))
```

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

In [21]:
```
ratings.shape
```

Out[21]:
```
(20000263, 4)
```

In [22]:
```
ratings.head(1)
```

Out[22]:

	userId	movieId	rating	timestamp
0	1	2	3.5	2005-04-02 23:53:47

In [23]:
```
tags=pd.read_csv(r"C:\Users\HP\OneDrive\Downloads\archive (1)\tag.csv")
```

In [24]:
```
print(type(tags))
```

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

In [25]:
```
tags.shape
```

Out[25]:
```
(465564, 4)
```

In [26]:
```
tags.head(1)
```

Out[26]:

	userId	movieId	tag	timestamp
0	18	4141	Mark Waters	2009-04-24 18:19:40

In [27]:
```
tags.tail(1)
```

Out[27]:

	userId	movieId	tag	timestamp
465563	138472	923	rise to power	2007-11-02 21:12:47

In [28]:
```
movies=pd.read_csv(r"C:\Users\HP\OneDrive\Downloads\archive (1)\movie.csv")
```

In [29]:
```
print(type(movies))
```

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

In [30]:
```
movies.shape
```

Out[30]:
```
(27278, 3)
```

In [31]:
```
movies.head(1)
```

Out[31]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy

In [32]:
```
del ratings['timestamp']# for current analysis we are deleting timestamp
del tags['timestamp']
```

In [30]:
```
ratings
```

Out[30]:

	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 [47]:
```
tags
```

Out[47]:

	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 Structures

SERIES

In [34]:
```
row_0=tags.iloc[0]
type(row_0)
```

Out[34]:
```
pandas.core.series.Series
```

In [38]:
```
row_0
```

Out[38]:
```
userId                        18
movieId                     4141
tag                  Mark Waters
timestamp    2009-04-24 18:19:40
Name: 0, dtype: object
```

In [40]:
```
row_0.index
```

Out[40]:
```
Index(['userId', 'movieId', 'tag', 'timestamp'], dtype='object')
```

In [42]:
```
row_0['userId']
```

Out[42]:
```
18
```

In [46]:
```
'rating' in row_0
```

Out[46]:
```
False
```

In [48]:
```
row_0.name
```

Out[48]:
```
0
```

In [52]:
```
row_0=row_0.rename('FirstRow')
row_0.name
```

Out[52]:
```
'FirstRow'
```

DataFrames

In [56]:
```
tags.head()
```

Out[56]:

	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 [58]:
```
tags.index
```

Out[58]:
```
RangeIndex(start=0, stop=465564, step=1)
```

In [62]:
```
tags.columns
```

Out[62]:
```
Index(['userId', 'movieId', 'tag', 'timestamp'], dtype='object')
```

In [51]:
```
tags.iloc[[0,856,12356]]
```

Out[51]:

	userId	movieId	tag
0	18	4141	Mark Waters
856	359	26840	Takeshi Definitely Beat This Genre To The Ground
12356	2299	1261	tense

Descriptive Statistics

- how ratings are distributed

In [70]:
```
ratings['rating'].describe()
```

Out[70]:
```
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 [76]:
```
ratings.describe()
```

Out[76]:

	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 [54]:
```
ratings['rating'].std()
```

Out[54]:
```
1.051988919275684
```

In [56]:
```
ratings['rating'].min()
```

Out[56]:
```
0.5
```

In [58]:
```
ratings['rating'].mode()
```

Out[58]:
```
0    4.0
Name: rating, dtype: float64
```

In [70]:
```
ratings.corr()    #corr() calculates correlation bet columns in dataframe
```

Out[70]:

	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 [71]:
```
ratings['userId'].max()
```

```
Out[71]: 138493
```

```
In [72]: ratings.mean()
```

```
Out[72]: userId     69045.872583
         movieId     9041.567330
         rating         3.525529
         dtype: float64
```

```
In [90]: filter=ratings['rating']>0
         print(filter)
```

```
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 [92]: filter.all()
```

```
Out[92]: True
```

```
In [80]: 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[80]: False
```

```
In [94]: filter1.all()
```

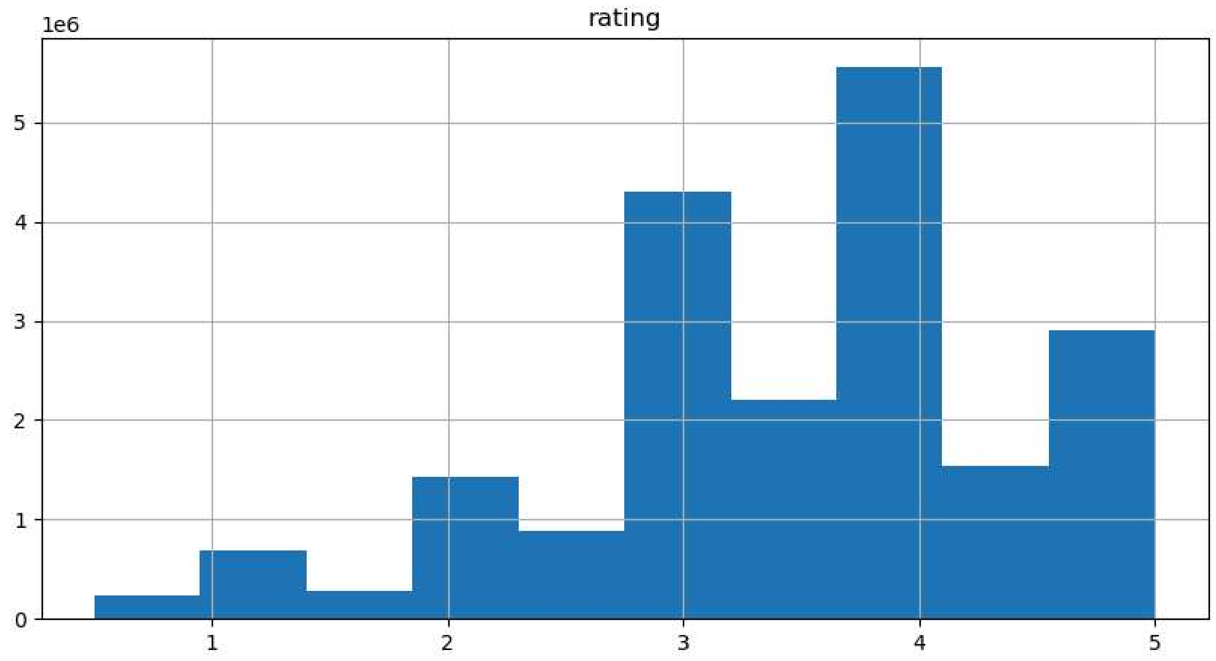
```
Out[94]: False
```

Data Cleaning:Handling missing values

In [100…
```
movies.shape
```
Out[100…
```
(27278, 3)
```
In [103…
```
movies.isnull().any()
```
Out[103…
```
movieId    False
title      False
genres     False
dtype: bool
```
In [105…
```
ratings.shape
```
Out[105…
```
(20000263, 3)
```
In [111…
```
ratings.isnull().any().any()
```
Out[111…
```
False
```
In [113…
```
tags.shape
```
Out[113…
```
(465564, 3)
```
In [123…
```
tags.isnull().any().any()  #prints True means we have some null values in tags
```
Out[123…
```
True
```
In [125…
```
tags=tags.dropna()
```
In [127…
```
tags.isnull().any().any()
```
Out[127…
```
False
```
In [131…
```
tags.shape  #no NULL values! number of lines have reduced
```
Out[131…
```
(465548, 3)
```

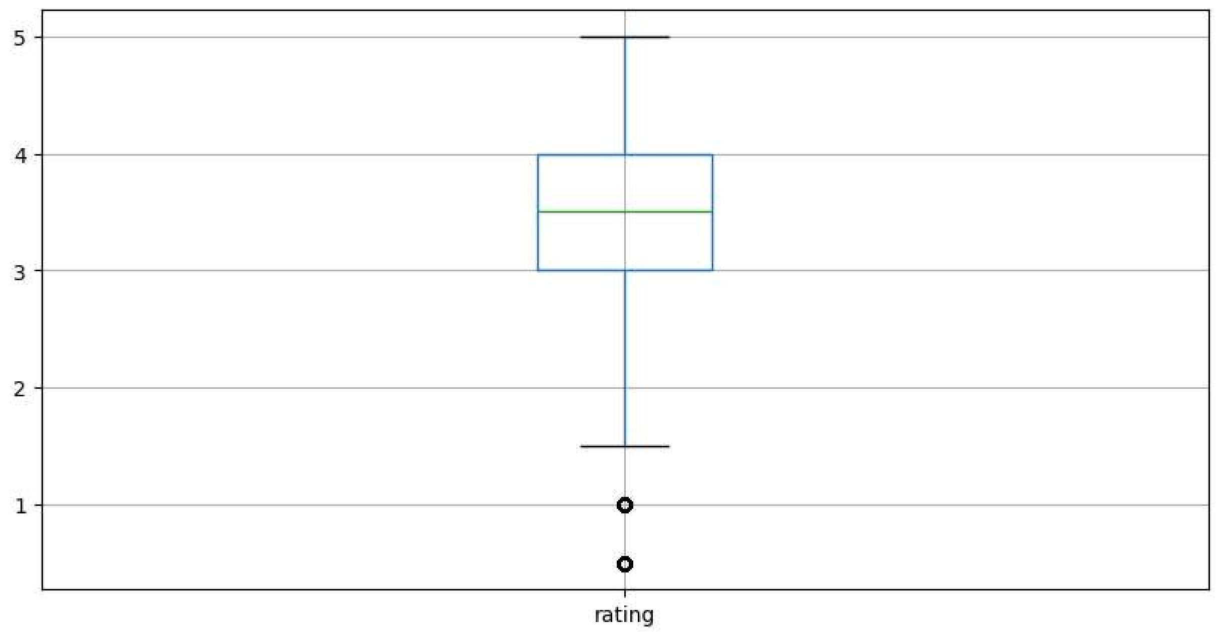
Data Visualization

In [134…
```
%matplotlib inline
import matplotlib.pyplot as plt
ratings.hist(column='rating',figsize=(10,5))
plt.show()
```



In [136…

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



Slicing Out Columns

In [139…

```
tags['tag'].head()
```

Out[139…

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

In [141…
```
movies[['title','genres']].head()
```

Out[141…

	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 [143…
```
ratings[-10:]
```

Out[143…

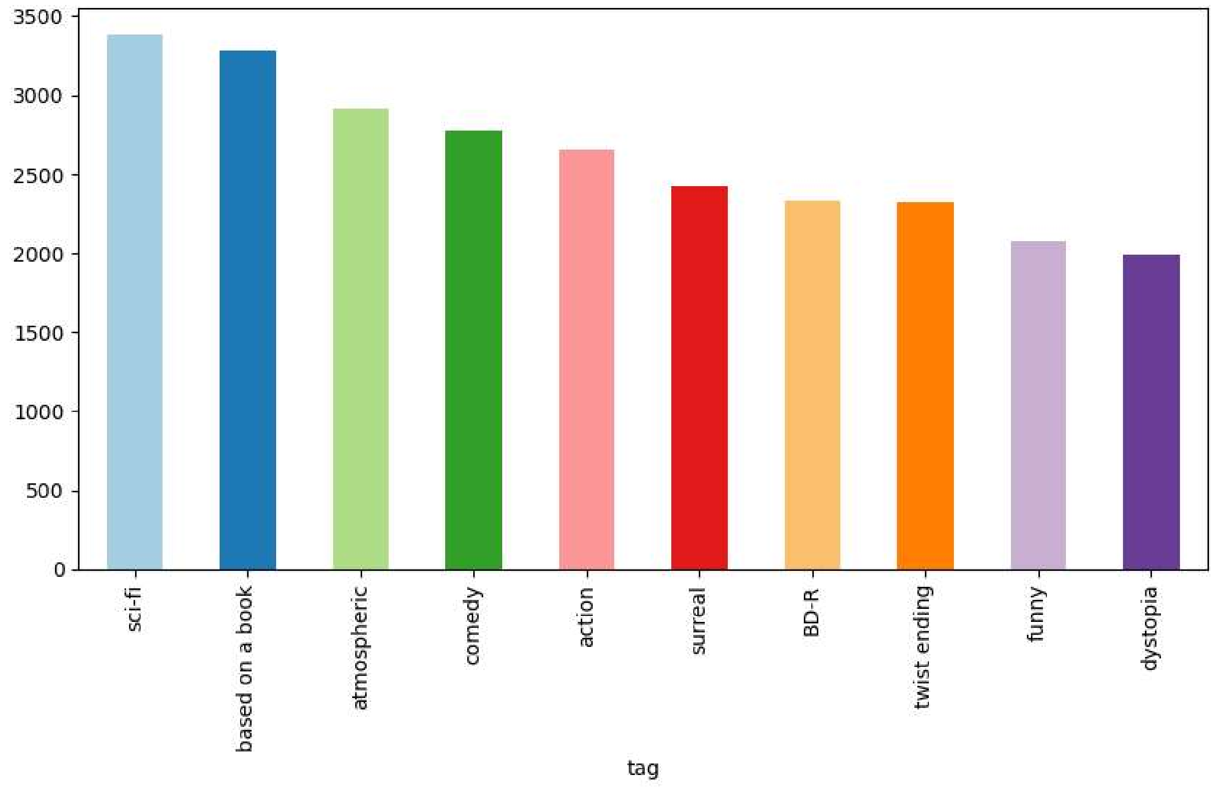
	userId	movieId	rating
20000253	138493	60816	4.5
20000254	138493	61160	4.0
20000255	138493	65682	4.5
20000256	138493	66762	4.5
20000257	138493	68319	4.5
20000258	138493	68954	4.5
20000259	138493	69526	4.5
20000260	138493	69644	3.0
20000261	138493	70286	5.0
20000262	138493	71619	2.5

In [145…
```
tags_count=tags['tag'].value_counts()
tags_count[-10:]
```

Out[145…
```
tag
missing child                    1
Ron Moore                        1
Citizen Kane                     1
mullet                           1
biker gang                       1
Paul Adelstein                   1
the wig                          1
killer fish                      1
genetically modified monsters    1
topless scene                    1
Name: count, dtype: int64
```

In [151…
```
colors=plt.cm.Paired.colors
tags_count[:10].plot(kind='bar',figsize=(10,5),color=colors)
```

```
Out[151…  <Axes: xlabel='tag'>
```


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 [ ]:
```