Pandas

Pandas Pandas သည် ဒေတာများကို လေ့လာရန်၊ analysis လုပ်ရန် အထူးပြုလုပ်ထားသည့် Python library တစ်ခုဖြစ်သည်။ ဒေတာများကို DataFrame အဖြစ် ပြုလုပ်ပြီး လေ့လာသည်။ ဒေတာများတည်ဆောက်ထားပုံကို data structure ဟုခေါ်သည်။ pandas DataFrame သည် Excel spreadsheet မှာ ဇယားများနှင့် တူညီသောကြောင့် Excel ကို အသုံးပြုတတ်သူအားလုံး အလွယ်တကူ လျှင်မြန်စွာလေ့လာ သင်ယူနိုင်သည်။ Excel spreadsheet ၏ ဒေတာတည်ဆောက်ထားပုံ(data structure) ဇယားပုံစံ (table format) ဖြစ်သည်။ ထို့အပြင် SQL မှကဲ့သို့ ဇယားများကို query လုပ်ခြင်း၊ ဆက်ခြင်း (SQL-like queries and joins of tables) စသည်တို့ ပြုလုပ်နိုင်သည်။

Pandas နှင့် NumPy မတူညီသည့်အချက်မှာ NumPy တွင် ထည့်သွင်းရသည့် အရာအားလုံး(NumPy တွင် ရှိသည့် အရာအားလုံး)သည် ဒေတာအမျိုးအစားတူညီကြသည့် array (array be of the same type) များ ဖြစ်ကြသည်။ Pandas တွင် ဒေတာများကို ဇယားပုံစံ (table format) ဖြင့် သိမ်းဆည်းထားသောကြောင့် ကော်လံတစ်ခုစီတွင် မတူညီသည် data format များဖြစ်သည့် integers ၊ dates ၊ floating-point numbers ၊ strings စသည်တို့ ဖြစ်နိုင်သည်။ တစ်နည်းအားဖြင့် NumPy သည် data format တူညီသည့် array များကို ကိုင်တွယ် စီမံပေးသည့်ရန် အတွက်ဖြစ်သည်။ Pandas သည် data format မတူညီသည့် ဇယားပုံစံ (table format) ဒေတာများကို လေ့လာနိုင်ရန်အတွက် ဖြစ်သည်။

Pandas သည် အမျိုးမျိုးသော ဒေတာဖိုင်များနှင့် data- bases များ အားလုံးကို ဖတ်ယူနိုင်စွမ်းရှိသည်။ ဥပမာ SQL database ၊ Excel ဖိုင်၊ comma-separated values (CSV)ဖိုင် တို့ကို အလွယ်တကူ ဖတ်ယူနိုင်သည်။ Pandas သည် machine learning ဘာသာရပ်တွင် ဒေတာများ သန့်စင်ခြင်း(cleaning) data ၊ exploration လုပ်ခြင်း နှင့် transformation လုပ်ခြင်းတို့အတွက် အလွန်အသုံးဝင်သည်။ data manipulation function များစွာပါဝင်သည်။ indexing အားသာချက်များစွာ ရှိသည်။

- data များ၏ statistics အချက်အလက်များကို အလွယ်တကူ ရှာဖွေနိုင်သည်။
- Data cleaning လုပ်ရန်အတွက် built in pandas function များစွာ ရှိသည်။
- ဒေတာများ subsetting လုပ်ရန် ၊ filteringလုပ်ရန် ၊ insertion လုပ်ရန် ၊ deletion လုပ်ရန် နှင့် aggregation လုပ်ရန် function များစွာ ရှိသည်။
- data set များ ပေါင်းခြင်း(Merging)၊ ဆက်ခြင်း ပြုလုပ်ရန် function များစွာ ရှိသည်။
- time-series data များအတွက် timestamps များ ၊ အချိန်နှင့် သက်ဆိုင်သည့် function များစွာ ရှိသည်။

Additional Recommended Resources:

- *pandas* Documentation: http://pandas.pydata.org/pandas-docs/stable/ (http://pandas.pydata.org/pandas-docs/stable/)
- *Python for Data Analysis* by Wes McKinney
- *Python Data Science Handbook* by Jake VanderPlas

Import Libraries

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

Introduction to pandas Data Structures

pandas ၏ အဓိက ဒေတာတည်ဆောက်ပုံနှစ်မျိုး(two main data structures)မှာ *Series* နှင့် *DataFrames* တို့ ဖြစ်သည်။

pandas Series

pandas Series သည် one-dimensional labeled array ဖြစ်သည်။ `pd.Series()` ဖြင့် pandas series တစ်ခုတည်ဆောက်သည်။

```
In [2]: ser = pd.Series([100, 'foo', 300, 'bar', 500], ['tom', 'bob', 'nancy
        ', 'dan', 'eric'])
```

တည်ဆောက်ပြီးသည့် pandas series ကို ပြန်ကြည့်ရန်

```
In [3]: ser
```

```
Out[3]: tom      100
        bob      foo
        nancy    300
        dan      bar
        eric     500
        dtype: object
```

pandas series မှ index များကို ပြန်ကြည့်ရန်

```
In [4]: ser.index
```

```
Out[4]: Index(['tom', 'bob', 'nancy', 'dan', 'eric'], dtype='object')
```

pandas series မှ axes ကို ကြည့်ရန် (pandas series တွင် axis တစ်ခုသာ ရှိသောကြောင့် axis နှင့် index သည် အတူတူပင်ဖြစ်သည်။)

```
In [5]: ser.axes
```

```
Out[5]: [Index(['tom', 'bob', 'nancy', 'dan', 'eric'], dtype='object')]
```

pandas series မှ 'nancy' နှင့် 'bob' တို့၏ သက်ဆိုင်ရာ တန်ဖိုးများကို `ser.loc[ ]` ဖြင့် access လုပ်နိုင်သည်။ pandas series ၏ index မှ 'nancy' ၊ 'bob' စသည် နာမည်များကို ပေး၍ access လုပ်နိုင်သည်။

```
In [6]: ser.loc[['nancy','bob']]
```

```
Out[6]: nancy    300
        bob      foo
        dtype: object
```

pandas series ၏ index မှ နံပါတ်(row number) ပေး၍ access လုပ်နိုင်သည်။

```
In [7]: ser[[4, 3, 1]]
```

```
Out[7]: eric    500
        dan     bar
        bob     foo
        dtype: object
```

pandas series ၏ index မှ နံပါတ်(row number) ပေး၍ `.iloc[ ]` ဖြင့် access လုပ်နိုင်သည်။

```
In [8]: ser.iloc[2]
```

```
Out[8]: 300
```

pandas series ၏ index မှ 'nancy' ၊ 'bob' စသည် နာမည်များကို ပေး၍ series ထဲတွင် ရှိ မရှိ စစ်နိုင်သည်။ ရှိနေလျှင် True သို့မဟုတ် မရှိလျှင် False ဖြင့် ဖော်ပြလိမ့်မည်။

```
In [9]: 'bob' in ser
```

```
Out[9]: True
```

pandas series ၏ နာမည်ဖြင့် ထို series ထဲတွင် ရှိနေသည့် item များ အားလုံးကို ကြည့်နိုင်သည်။

```
In [10]: ser
```

```
Out[10]: tom      100
         bob      foo
         nancy    300
         dan      bar
         eric     500
         dtype: object
```

pandas series တစ်ခုလုံးကို ၂ နှင့်မြှောက်လျှင် item သည် တန်ဖိုးများ(value)ဖြစ်နေလျှင် ၂ဆတိုးပြီး string ဖြစ်နေလျှင် (၂)ခုဖြစ်အောင် လုပ်ပေးလိမ့်မည်။

ser * 2

pandas series အတွင်းမှ item များကို နှစ်ထပ်ကိန်းတင်လျှင်ခြင်း

```
In [11]: ser[['nancy', 'eric']] ** 2
```

```
Out[11]: nancy     90000
         eric     250000
         dtype: object
```

pandas DataFrame

pandas DataFrame သည် 2-dimensional labeled data structure တစ်ခုဖြစ်သည်။

Create DataFrame from dictionary of Python Series

pandas series နှစ်ခုဖြင့် DataFrame တစ်ခု တည်ဆောက်သည်။ DataFrame ဖြစ်သောကြောင့် row index နှင့် column index ဟူ၍ index (၂)ခုရှိသည်။

In [12]:
```
d = {'one' : pd.Series([100., 200., 300.], index=['apple', 'ball', 'clock']),
     'two' : pd.Series([111., 222., 333., 4444.], index=['apple', 'ball', 'cerill', 'dancy'])}
```

d သည် DataFrame တစ်ခု မဖြစ်သေးပါ။

In [13]:
```
print(type(d))
```

```
<class 'dict'>
```

In [14]:
```
d
```

Out[14]:
```
{'one': apple    100.0
 ball     200.0
 clock    300.0
 dtype: float64, 'two': apple      111.0
 ball       222.0
 cerill     333.0
 dancy     4444.0
 dtype: float64}
```

`pd.DataFrame(d)` ဖြင့် DataFrame ဖြစ်အောင် ပြုလုပ်သည်။ DataFrame ဖြစ်သည့်အခါ index များကို ဘုံဖြစ်အောင်(common) ပြုလုပ်ပြီး မရှိသည့် တန်ဖိုးများနေရာတွင် NaN ဖြင့် အစားထိုးသည်။

In [15]:
```
df = pd.DataFrame(d)
print(df)
```

```
          one     two
apple   100.0   111.0
ball    200.0   222.0
cerill    NaN   333.0
clock   300.0     NaN
dancy     NaN  4444.0
```

DataFrame တွင် index သည် row index ဖြစ်သည်။ column index သည် column ဖြင့်သာ ရေးပေးရန်လိုသည်။

In [16]:
```
df.index
```

Out[16]:
```
Index(['apple', 'ball', 'cerill', 'clock', 'dancy'], dtype='object')
```

In [17]:
```
df.columns
```

Out[17]:
```
Index(['one', 'two'], dtype='object')
```

အထက်တွင် တည်ဆောက်ခဲ့သည် d ဆိုသည့် dict မှ မိမိအလိုရှိသည့် item များကို သာယူ၍ DataFrame တစ်ခုကို အောက်ပါအတိုင်း တည်ဆောက်နိုင်သည်။

```
In [18]: pd.DataFrame(d, index=['dancy', 'ball', 'apple'])
```

Out[18]:

	one	two
dancy	NaN	4444.0
ball	200.0	222.0
apple	100.0	111.0

အထက်တွင် တည်ဆောက်ခဲ့သည် d ဆိုသည့် dict မှ မိမိအလိုရှိသည့် item များကို သာယူ၍ DataFrame တစ်ခုကို အောက်ပါအတိုင်း တည်ဆောက်နိုင်သလို မိမိအလိုရှိသည့် ကော်လံကိုလည်း မိမိကြိုက်သည့် ကော်လံနာမည်ပေးနိုင်သည်။ သို့သော် သက်ဆိုင်သည်တန်ဖိုးများ မရှိပါက NaN ဖြင့် အစားထိုးပေးလိမ့်မည်။

```
In [19]: pd.DataFrame(d, index=['dancy', 'ball', 'apple'], columns=['two', 'five'])
```

Out[19]:

	two	five
dancy	4444.0	NaN
ball	222.0	NaN
apple	111.0	NaN

Create DataFrame from list of Python dictionaries

Python dictionary မှ DataFrame တစ်ခု တည်ဆောက်ခြင်း

```
In [20]: data = [{'alex': 1, 'joe': 2}, {'ema': 5, 'dora': 10, 'alice': 20}]
```

```
In [21]: pd.DataFrame(data)
```

Out[21]:

	alex	alice	dora	ema	joe
0	1.0	NaN	NaN	NaN	2.0
1	NaN	20.0	10.0	5.0	NaN

```
In [22]: pd.DataFrame(data, index=['orange', 'red'])
```

Out[22]:

	alex	alice	dora	ema	joe
orange	1.0	NaN	NaN	NaN	2.0
red	NaN	20.0	10.0	5.0	NaN

In [23]:
```
pd.DataFrame(data, columns=['joe', 'dora','alice'])
```

Out[23]:

	joe	dora	alice
0	2.0	NaN	NaN
1	NaN	10.0	20.0

Basic DataFrame operations

In [24]:
```
df
```

Out[24]:

	one	two
apple	100.0	111.0
ball	200.0	222.0
cerill	NaN	333.0
clock	300.0	NaN
dancy	NaN	4444.0

ကော်လံနာမည်ပေး၍ DataFrame တစ်ခုမှ ထိုကော်လံအတွင်းရှိ item များကို access လုပ်ခြင်း

In [25]:
```
df['one']
```

Out[25]:
```
apple     100.0
ball      200.0
cerill      NaN
clock     300.0
dancy       NaN
Name: one, dtype: float64
```

ကော်လံ 'one' နှင့် ကော်လံ 'two' ကို မြှောက်၍ ကော်လံ 'three' နာမည်ဖြင့် ကော်လံ အသစ်တစ်ခု ထပ်ထည့်ခြင်း

In [26]:
```
df['three'] = df['one'] * df['two']
df
```

Out[26]:

	one	two	three
apple	100.0	111.0	11100.0
ball	200.0	222.0	44400.0
cerill	NaN	333.0	NaN
clock	300.0	NaN	NaN
dancy	NaN	4444.0	NaN

ကော်လံ 'one' အတွင်းမှ item များသည် 250 ထက်ပိုများလျှင် True သို့မဟုတ် ပိုနည်းလျှင် False ဆိုသည့် condition ဖြင့် 'flag' ကော်လံတစ်ခု ထပ်ထည့်ခြင်း

In [27]:
```
df['flag'] = df['one'] > 250
df
```

Out[27]:

	one	two	three	flag
apple	100.0	111.0	11100.0	False
ball	200.0	222.0	44400.0	False
cerill	NaN	333.0	NaN	False
clock	300.0	NaN	NaN	True
dancy	NaN	4444.0	NaN	False

In [28]:
```
three = df.pop('three')
```

In [29]:
```
three
```

Out[29]:
```
apple     11100.0
ball      44400.0
cerill        NaN
clock         NaN
dancy         NaN
Name: three, dtype: float64
```

In [30]:
```
df
```

Out[30]:

	one	two	flag
apple	100.0	111.0	False
ball	200.0	222.0	False
cerill	NaN	333.0	False
clock	300.0	NaN	True
dancy	NaN	4444.0	False

In [31]:
```
del df['two']
```

In [32]:
```
df
```

Out[32]:

	one	flag
apple	100.0	False
ball	200.0	False
cerill	NaN	False
clock	300.0	True
dancy	NaN	False

In [33]:
```
df.insert(2, 'copy_of_one', df['one'])
df
```

Out[33]:

	one	flag	copy_of_one
apple	100.0	False	100.0
ball	200.0	False	200.0
cerill	NaN	False	NaN
clock	300.0	True	300.0
dancy	NaN	False	NaN

In [34]:
```
df['one_upper_half'] = df['one'][:2]
df
```

Out[34]:

	one	flag	copy_of_one	one_upper_half
apple	100.0	False	100.0	100.0
ball	200.0	False	200.0	200.0
cerill	NaN	False	NaN	NaN
clock	300.0	True	300.0	NaN
dancy	NaN	False	NaN	NaN

Case Study: Movie Data Analysis

This notebook uses a dataset from the MovieLens website. We will describe the dataset further as we explore with it using *pandas*.

Download the Dataset

Please note that **you will need to download the dataset**. Although the video for this notebook says that the data is in your folder, the folder turned out to be too large to fit on the edX platform due to size constraints.

Here are the links to the data source and location:

- **Data Source:** MovieLens web site (filename: ml-20m.zip)
- **Location:** https://grouplens.org/datasets/movielens/ (https://grouplens.org/datasets/movielens/)

Use Pandas to Read the Dataset

In this notebook, we will be using three CSV files:

- **ratings.csv :** *userId*,*movieId*,*rating*, *timestamp*
- **tags.csv :** *userId*,*movieId*, *tag*, *timestamp*
- **movies.csv :** *movieId*, *title*, *genres*

Using the *read_csv* function in pandas, we will ingest these three files.

movies.csv ကို ဖတ်၍ ဒေတာများကို ယူသည်။

In [35]:
```
movies = pd.read_csv('movies.csv', sep=',')
print(type(movies))
movies.head(15)
```

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

Out[35]:

	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

tags.csv ကို ဖတ်၍ ဒေတာများကို ယူသည်။

In [36]:
```
# Timestamps represent seconds since midnight Coordinated Universal
Time (UTC) of January 1, 1970
tags = pd.read_csv('tags.csv', sep=',')
tags.head()
```

Out[36]:

	userId	movieId	tag	timestamp
0	3	260	classic	1439472355
1	3	260	sci-fi	1439472256
2	4	1732	dark comedy	1573943598
3	4	1732	great dialogue	1573943604
4	4	7569	so bad it's good	1573943455

ratings.csv ကို ဖတ်၍ ဒေတာများကို ယူသည်။

In [37]:
```
ratings = pd.read_csv('ratings.csv', sep=',', parse_dates=['timestam
p'])
ratings.head()
```

Out[37]:

	userId	movieId	rating	timestamp
0	1	31	2.5	1260759144
1	1	1029	3.0	1260759179
2	1	1061	3.0	1260759182
3	1	1129	2.0	1260759185
4	1	1172	4.0	1260759205

ratings DataFrame မှ 'timestamp' ကော်လံကို ဖယ်ရှားသည်။ (ratings.csv မှ ဒေတာများကို တကယ် ဖျက်ပစ်ခြင်း မဟုတ်ပါ။)
tags DataFrame မှ 'timestamp' ကော်လံကို ဖယ်ရှားသည်။

In [38]:
```
# For current analysis, we will remove timestamp (we will come back
to it!)

del ratings['timestamp']
del tags['timestamp']
```

Data Structures

Series

tags DataFrame မှ ပထမဆုံး row (row နံပါတ် 0) ကို series တစ်ခု အဖြစ် ပြုလုပ်သည်။

In [39]:
```
#Extract 0th row: notice that it is infact a Series

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

Out[39]:
```
pandas.core.series.Series
```

In [40]:
```
print(row_0)
```

```
userId           3
movieId        260
tag        classic
Name: 0, dtype: object
```

index ကို access လုပ်သည်။

In [41]:
```
row_0.index
```

Out[41]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

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

Out[42]:
```
3
```

In [43]:
```
'rating' in row_0
```

Out[43]:
```
False
```

row_0 ၏ နာမည်ကို ဖတ်သည်။

In [44]:
```
row_0.name
```

Out[44]:
```
0
```

row_0 ၏ နာမည်ကို ၏ နာမည်ကို 'first_row' ဟု နာမည်ပြောင်းသည်။

In [45]:
```
row_0 = row_0.rename('first_row')
row_0.name
```

Out[45]:
```
'first_row'
```

DataFrames

tags DataFrame ၏ ပထမဆုံး row (၅)ခုကို ကြည့်ရန်

In [46]:
```
tags.head()
```

Out[46]:

	userId	movieId	tag
0	3	260	classic
1	3	260	sci-fi
2	4	1732	dark comedy
3	4	1732	great dialogue
4	4	7569	so bad it's good

tags DataFrame ၏ row index များကို ကြည့်ရန်

In [47]:
```
tags.index
```

Out[47]:
```
RangeIndex(start=0, stop=1093360, step=1)
```

tags DataFrame ၏ ကော်လံ index များကို ကြည့်ရန်

In [48]:
```
tags.columns
```

Out[48]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

row index နံပါတ် 0 ၊ 11 နှင့် 2000 တို့၏ ဒေတာများကို ကြည့်ရန်

In [49]:
```
# Extract row 0, 11, 2000 from DataFrame

tags.iloc[ [0,11,2000] ]
```

Out[49]:

	userId	movieId	tag
0	3	260	classic
11	4	164909	cliche
2000	647	164179	twist ending

Descriptive Statistics

ratings dataframe မှ rating ကော်လံ၏ အချက်အလက်များ

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

```
Out[50]: count    100004.000000
         mean          3.543608
         std           1.058064
         min           0.500000
         25%           3.000000
         50%           4.000000
         75%           4.000000
         max           5.000000
         Name: rating, dtype: float64
```

ratings dataframe မှ အချက်အလက်များကို ကြည့်ရန်။ userId ၊ movieId ၊ rating စသည့် ကော်လံများ အားလုံး၏ အချက်အလက်များကို ဖော်ပြပေးသည်။

```
In [51]: ratings.describe()
```

Out[51]:

	userId	movieId	rating
count	100004.000000	100004.000000	100004.000000
mean	347.011310	12548.664363	3.543608
std	195.163838	26369.198969	1.058064
min	1.000000	1.000000	0.500000
25%	182.000000	1028.000000	3.000000
50%	367.000000	2406.500000	4.000000
75%	520.000000	5418.000000	4.000000
max	671.000000	163949.000000	5.000000

ratings dataframe မှ rating ကော်လံ၏ mean တန်ဖိုးတစ်ခုတည်းကိုသာ ဖော်ပြရန်

```
In [52]: ratings['rating'].mean()
```

```
Out[52]: 3.543608255669773
```

ratings dataframe မှ ကော်လံများအားလုံး၏ mean တန်ဖိုးကို ဖော်ပြရန်

```
In [53]: ratings.mean()
```

```
Out[53]: userId       347.011310
         movieId    12548.664363
         rating         3.543608
         dtype: float64
```

အနည်းဆုံး တန်ဖိုးကို ဖော်ပြရန်

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

```
Out[54]: 0.5
```

အများဆုံး တန်ဖိုးကို ဖော်ပြရန်

```
In [55]: ratings['rating'].max()
```

```
Out[55]: 5.0
```

std တန်ဖိုးကို ဖော်ပြရန်

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

```
Out[56]: 1.0580641091073735
```

အကြိမ်အများဆုံးဖြစ်သည့် ကိန်းတန်ဖိုးကို ဖော်ပြရန်

```
In [57]: ratings['rating'].mode()
```

```
Out[57]: 0    4.0
         dtype: float64
```

Correlation matrix ကို ဖော်ပြရန် ။ pairwise correlation of columns ကို တွက်ပေးသည်။ missing value များကို ထည့်မတွက်ပါ။ (excluding NA/null values.)

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

Out[58]:

	userId	movieId	rating
userId	1.000000	0.007126	0.010467
movieId	0.007126	1.000000	-0.028894
rating	0.010467	-0.028894	1.000000

ကော်လံ 'rating' အတွင်းမှ item များသည် 5 ထက်ပိုများလျှင် True သို့မဟုတ် ပိုနည်းလျှင် False ဖော်ပြရန်

In [59]:
```
filter_1 = ratings['rating'] > 5
print(filter_1)
filter_1.any()
```

```
0         False
1         False
2         False
3         False
4         False
5         False
6         False
7         False
8         False
9         False
10        False
11        False
12        False
13        False
14        False
15        False
16        False
17        False
18        False
19        False
20        False
21        False
22        False
23        False
24        False
25        False
26        False
27        False
28        False
29        False
          ...
99974     False
99975     False
99976     False
99977     False
99978     False
99979     False
99980     False
99981     False
99982     False
99983     False
99984     False
99985     False
99986     False
99987     False
99988     False
99989     False
99990     False
99991     False
99992     False
99993     False
99994     False
99995     False
99996     False
99997     False
99998     False
99999     False
100000    False
100001    False
100002    False
```

Out[59]: False

In [60]:
```
filter_2 = ratings['rating'] > 0
filter_2.all()
```

Out[60]: True

Data Cleaning: Handling Missing Data

movies DataFrame၏ row အရေအတွက် နှင့် ကော်လံအရေအတွက်ကို ဖော်ပြရန်

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

Out[61]: (9125, 3)

movies DataFrame တွင် missing value များ ရှိ မရှိစစ်ခြင်းဖြစ်သည်။
isnull() သည် missing value များ ရှိ မရှိစစ်ခြင်းဖြစ်သည်။ .any() သည် ရှိ မရှိကို ကော်လံအလိုက် ဖော်ပြပေးသည်။
isnull() ဖြင့် စစ်လျှင် DataFrame တစ်ခုလုံးကို ဖော်ပြသည်။

In [62]:
```
#is any row NULL ?

movies.isnull().any()
```

Out[62]:
```
movieId    False
title      False
genres     False
dtype: bool
```

ratings dataframe ၏ row အရေအတွက်နှင့် ကော်လံ အရေအတွက်ကို ဖော်ပြရန်

In [63]:
```
ratings.shape
```

Out[63]: (100004, 3)

ratings DataFrame တွင် missing value များ ရှိ မရှိစစ်ခြင်းဖြစ်သည်။ isnull() သည် missing value များ ရှိ မရှိစစ်ခြင်းဖြစ်သည်။ .any() သည် ရှိ မရှိကို ကော်လံအလိုက် ဖော်ပြပေးသည်။ isnull() ဖြင့် စစ်လျှင် DataFrame တစ်ခုလုံးကို ဖော်ပြသည်။

In [64]:
```
#is any row NULL ?

ratings.isnull().any()
```

Out[64]:
```
userId     False
movieId    False
rating     False
dtype: bool
```

Thats nice ! No NULL values !

In [65]:
```
tags.shape
```

Out[65]:
```
(1093360, 3)
```

In [66]:
```
#is any row NULL ?

tags.isnull().any()
```

Out[66]:
```
userId     False
movieId    False
tag         True
dtype: bool
```

tags dataframe တွင် missing value များရှိသည်။ ထို missing value များကို ဖယ်ထုတ်ရန် `.dropna()` ကို အသုံးပြုသည်။

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

In [68]:
```
#Check again: is any row NULL ?

tags.isnull().any()
```

Out[68]:
```
userId     False
movieId    False
tag        False
dtype: bool
```

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

Out[69]:
```
(1093344, 3)
```

ထို missing value များကို ဖယ်ထုတ်လိုက်သောကြောင့် row အရေအတွက် လျော့နည်းသွားသည်။ (No NULL values ! Notice the number of lines have reduced.)

Slicing Out Columns

tags dataframe မှ 'tag' ကော်လံ၏ ပထမဆုံး (၅)ခုကို ဖော်ပြရန်

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

Out[70]:
```
0              classic
1               sci-fi
2          dark comedy
3       great dialogue
4    so bad it's good
Name: tag, dtype: object
```

movies dataframe မှ 'title' နှင့် 'genres' ကော်လံတို့၏ ပထမဆုံး (၅)ခုကို ဖော်ပြရန်

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

ratings dataframe မှ နောက်ဆုံး (၁၀)ခုကို ဖော်ပြရန်

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

Out[72]:

	userId	movieId	rating
99994	671	5952	5.0
99995	671	5989	4.0
99996	671	5991	4.5
99997	671	5995	4.0
99998	671	6212	2.5
99999	671	6268	2.5
100000	671	6269	4.0
100001	671	6365	4.0
100002	671	6385	2.5
100003	671	6565	3.5

ratings dataframe မှ နောက်ဆုံး (၅)ခုကို ဖော်ပြရန်

In [73]:
```
ratings.tail()
```

Out[73]:

	userId	movieId	rating
99999	671	6268	2.5
100000	671	6269	4.0
100001	671	6365	4.0
100002	671	6385	2.5
100003	671	6565	3.5

tags datafram မှ 'tag' ကော်လံမှ ရုပ်ရှင်အမျိုးအစားများကို ရေတွက်ပြီး ပထမဆုံး (၅)ခုကို ဖော်ပြရန်

In [74]:
```
tag_counts = tags['tag'].value_counts()
tag_counts[:5]
```

Out[74]:
```
sci-fi          8330
atmospheric     6516
action          5907
comedy          5702
surreal         5326
Name: tag, dtype: int64
```

tags datafram မှ 'tag' ကော်လံမှ ရုပ်ရှင်အမျိုးအစားများကို ရေတွက်ပြီး နောက်ဆုံး(၁၀)ခုကို ဖော်ပြရန်

In [75]:
```
tag_counts = tags['tag'].value_counts()
tag_counts[-5:]
```

Out[75]:
```
Marcos Jorge                                1
Judith Ivey                                 1
there must be something clever here...      1
rich = corrupt                              1
pen to the neck                             1
Name: tag, dtype: int64
```

tags datafram မှ 'tag' ကော်လံမှ ရုပ်ရှင်အမျိုးအစားများကို ရေတွက်ပြီး အများဆုံး (၅)ခုကိုဘား ဂရပ်ဖြင့်ဖော်ပြရန်

In [76]:
```
tag_counts[:5].plot(kind='bar', figsize=(10,3))
```

Out[76]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xbeba3c8>
```

Filters for Selecting Rows

ratings datafram မှ 'rating' ကော်လံရှိတန်ဖိုးများ 4.0 သို့မဟုတ် 4.0 များလျှင် rating ကောင်းသည်(highly_rated)သတ်မှတ်သည်။ ထို rating ကောင်းသည်(highly_rated) ရုပ်ရှင်များထဲမှ ပထမဆုံး (၅)ကိုသာ ဖော်ပြရန်

In [77]:
```
is_highly_rated = ratings['rating'] >= 4.0

ratings[is_highly_rated][:5]
```

Out[77]:

	userId	movieId	rating
4	1	1172	4.0
12	1	1953	4.0
13	1	2105	4.0
20	2	10	4.0
21	2	17	5.0

movies datafram မှ 'genres' ကော်လံတွင် 'Animation' ဟုဖော်ထားလျှင် Animation ပါသည့် ရုပ်ရှင်ဖြစ်သည်။ ထို Animation ရုပ်ရှင်ကားများဟုတ် မဟုတ်စစ်ဆေးပြီး ရုပ်ရှင်များထဲမှ ပထမဆုံး (၅)ကိုသာ ဖော်ပြရန်

In [78]:
```
is_animation = movies['genres'].str.contains('Animation')
movies[is_animation].head(5)
```

Out[78]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
12	13	Balto (1995)	Adventure|Animation|Children
46	48	Pocahontas (1995)	Animation|Children|Drama|Musical|Romance
211	239	Goofy Movie, A (1995)	Animation|Children|Comedy|Romance
216	244	Gumby: The Movie (1995)	Animation|Children

Group By and Aggregate

In [79]:
```
ratings[['movieId','rating']].head(5)
```

Out[79]:

	movieId	rating
0	31	2.5
1	1029	3.0
2	1061	3.0
3	1129	2.0
4	1172	4.0

'rating' ကော်လံမှ တန်ဖိုးများ တူရာစုပြီး ရေတွက်ပါ။ .groupby('rating') သည် 'rating' ကော်လံမှ တန်ဖိုးများ တူရာစုခြင်းဖြစ်သည်။ ဥပမာ rating = 3.0 ဖြစ်သည့် ရုပ်ရှင်ကားများ အားလုံးကို စုပေါင်း၍ ဘယ်နှစ်ခုရှိသည်ကို ရေတွက်သည်။

In [80]:
```
ratings_count = ratings[['movieId','rating']].groupby('rating').coun
t()
ratings_count
```

Out[80]:

rating	movieId
0.5	1101
1.0	3326
1.5	1687
2.0	7271
2.5	4449
3.0	20064
3.5	10538
4.0	28750
4.5	7723
5.0	15095

'movieId' ကော်လံမှ တန်ဖိုးများ တူရာစုပြီး ပျှမ်းမျှတန်ဖိုး(mean) ရှာသည်။

In [81]:
```
average_rating = ratings[['movieId','rating']].groupby('movieId').me
an()
average_rating.head()
```

Out[81]:

movieId	rating
1	3.872470
2	3.401869
3	3.161017
4	2.384615
5	3.267857

'movieId' ကော်လံမှ တန်ဖိုးများ တူရာစုပြီး ရေတွက်သည်။

In [82]:
```
movie_count = ratings[['movieId','rating']].groupby('movieId').count
()
movie_count.head()
```

Out[82]:

movieId	rating
1	247
2	107
3	59
4	13
5	56

In [83]:
```
movie_count = ratings[['movieId','rating']].groupby('movieId').count
()
movie_count.tail()
```

Out[83]:

movieId	rating
161944	1
162376	1
162542	1
162672	1
163949	1

Dataframe များပေါင်းစည်းခြင်း(Merge Dataframes)

In [84]:
```
tags.head()
```

Out[84]:

	userId	movieId	tag
0	3	260	classic
1	3	260	sci-fi
2	4	1732	dark comedy
3	4	1732	great dialogue
4	4	7569	so bad it's good

In [85]:
```
movies.head()
```

Out[85]:

	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 [86]:
```
tags.head(2)
```

Out[86]:

	userId	movieId	tag
0	3	260	classic
1	3	260	sci-fi

In [87]:
```
movies.head(2)
```

Out[87]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy

movies dataframe နှင့် tags dataframe ကိုပေါင်းမည်။ မည်သည့်နည်းဖြင့် ပေါင်းထည့်မည်ဆိုသည်ကို ပြောပေးရန်လိုသည်။ `how='inner'` ဆိုသည်မှာ dataframe နှစ်ခု စလုံးမှ key များကို intersection လုပ်သည့်နည်းဖြင့် ပေါင်းခြင်းဖြစ်သည်။ SQL မှ inner join နှင့် တူသည်။

inner: use intersection of keys from both frames, similar to a SQL inner join; preserve the order of the left keys.

In [88]:
```
t = movies.merge(tags, on='movieId', how='inner')
t.head()
```

Out[88]:

	movieId	title	genres	userId	tag
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	791	Owned
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1048	imdb top 250
2	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1361	Pixar
3	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	3164	Pixar
4	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	3164	time travel

More examples: http://pandas.pydata.org/pandas-docs/stable/merging.html (http://pandas.pydata.org/pandas-docs/stable/merging.html)

Combine aggreagation, merging, and filters to get useful analytics

In [89]:
```
avg_ratings = ratings.groupby('movieId', as_index=False).mean()
avg_ratings.head()
```

Out[89]:

	movieId	userId	rating
0	1	338.558704	3.872470
1	2	318.906542	3.401869
2	3	374.423729	3.161017
3	4	355.538462	2.384615
4	5	320.785714	3.267857

'userId' ကော်လံကို ဖယ်ထုတ်သည်။

In [90]:
```
avg_ratings = ratings.groupby('movieId', as_index=False).mean()
del avg_ratings['userId']
avg_ratings.head()
```

Out[90]:

	movieId	rating
0	1	3.872470
1	2	3.401869
2	3	3.161017
3	4	2.384615
4	5	3.267857

movies dataframe နှင့် အထက်က avg_ratings dataframe ကိုပေါင်းသည်။ merge လုပ်ပြီး box_office ဆိုသည့် dataframe တစ်ခုတည်ဆောက်သည်။

In [91]:
```
box_office = movies.merge(avg_ratings, on='movieId', how='inner')
box_office.tail()
```

Out[91]:

	movieId	title	genres	rating
9061	161944	The Last Brickmaker in America (2001)	Drama	5.0
9062	162376	Stranger Things	Drama	4.5
9063	162542	Rustom (2016)	Romance|Thriller	5.0
9064	162672	Mohenjo Daro (2016)	Adventure|Drama|Romance	3.0
9065	163949	The Beatles: Eight Days a Week - The Touring Y...	Documentary	5.0

rating မြင့်သည့် ရုပ်ရှင်ကားများကို စစ်ထုတ်သည်။

In [92]:
```
is_highly_rated = box_office['rating'] >= 4.0

box_office[is_highly_rated][-5:]
```

Out[92]:

	movieId	title	genres	rating
9055	160718	Piper (2016)	Animation	4.0
9061	161944	The Last Brickmaker in America (2001)	Drama	5.0
9062	162376	Stranger Things	Drama	4.5
9063	162542	Rustom (2016)	Romance|Thriller	5.0
9065	163949	The Beatles: Eight Days a Week - The Touring Y...	Documentary	5.0

Comedy ရုပ်ရှင်ကားများကို စစ်ထုတ်သည်။

In [93]:
```
is_comedy = box_office['genres'].str.contains('Comedy')

box_office[is_comedy][:5]
```

Out[93]:

	movieId	title	genres	rating
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	3.872470
2	3	Grumpier Old Men (1995)	Comedy|Romance	3.161017
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	2.384615
4	5	Father of the Bride Part II (1995)	Comedy	3.267857
6	7	Sabrina (1995)	Comedy|Romance	3.283019

rating မြင့်သည့် ဟာသ ရုပ်ရှင်ကား(comedy)များကိုသာ ဖော်ပြရန်

In [94]:
```
box_office[is_comedy & is_highly_rated][-5:]
```

Out[94]:

	movieId	title	genres	rating
9019	152081	Zootopia (2016)	Action|Adventure|Animation|Children|Comedy	4.0
9023	153584	The Last Days of Emma Blank (2009)	Comedy	5.0
9027	156025	Ice Age: The Great Egg-Scapade (2016)	Adventure|Animation|Children|Comedy	5.0
9037	158314	Daniel Tosh: Completely Serious (2007)	Comedy	4.5
9052	160567	Mike & Dave Need Wedding Dates (2016)	Comedy	4.0

Vectorized String Operations

In [95]:
```
movies.head()
```

Out[95]:

	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

Split 'genres' into multiple columns

'genres'ကော်လံမှ စာသားများကို ခွဲထုတ် ပစ်ရန်

In [96]:
```
movie_genres = movies['genres'].str.split('|', expand=True)
```

In [97]:
```
movie_genres[:10]
```

Out[97]:

	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
5	Action	Crime	Thriller	None	None	None	None	None	None	None
6	Comedy	Romance	None	None	None	None	None	None	None	None
7	Adventure	Children	None	None	None	None	None	None	None	None
8	Action	None	None	None	None	None	None	None	None	None
9	Action	Adventure	Thriller	None	None	None	None	None	None	None

Add a new column for comedy genre flag

ဟာသ ရုပ်ရှင်ကား (comedy)များအတွက် ဖော်ပြသည့် ကော်လံတစ်ခု ထပ်ထည့်ရန်

In [98]:
```
movie_genres['isComedy'] = movies['genres'].str.contains('Comedy')
```

In [99]:
```
movie_genres[:10]
```

Out[99]:

	0	1	2	3	4	5	6	7	8	9	isCo
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	
5	Action	Crime	Thriller	None	None	None	None	None	None	None	
6	Comedy	Romance	None	None	None	None	None	None	None	None	
7	Adventure	Children	None	None	None	None	None	None	None	None	
8	Action	None	None	None	None	None	None	None	None	None	
9	Action	Adventure	Thriller	None	None	None	None	None	None	None	

Extract year from title e.g. (1995)

'title' ကော်လံ မှ ရုပ်ရှင်ကားနာမည်ကို ဖတ်ပြီး ခုနှစ်များကို ယူ၍ Dataframe တွင် year ကော်လံတစ်ခုဖြင့် ထည့်၍ ဖော်ပြရန်

In [100]:
```
movies['year'] = movies['title'].str.extract('.*\((.*)\).*', expand=True)
```

In [101]:
```
movies.tail()
```

Out[101]:

	movieId	title	genres	year
9120	162672	Mohenjo Daro (2016)	Adventure|Drama|Romance	2016
9121	163056	Shin Godzilla (2016)	Action|Adventure|Fantasy|Sci-Fi	2016
9122	163949	The Beatles: Eight Days a Week - The Touring Y...	Documentary	2016
9123	164977	The Gay Desperado (1936)	Comedy	1936
9124	164979	Women of '69, Unboxed	Documentary	NaN

More here: http://pandas.pydata.org/pandas-docs/stable/text.html#text-string-methods

Parsing Timestamps

Timestamps are common in sensor data or other time series datasets. Let us revisit the *tags.csv* dataset and read the timestamps!

In [102]:
```
tags = pd.read_csv('tags.csv', sep=',')
```

In [103]:
```
tags.dtypes
```

Out[103]:
```
userId        int64
movieId       int64
tag          object
timestamp     int64
dtype: object
```

Unix time / POSIX time / epoch time records time in seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970

In [104]:
```
tags.head(5)
```

Out[104]:

	userId	movieId	tag	timestamp
0	3	260	classic	1439472355
1	3	260	sci-fi	1439472256
2	4	1732	dark comedy	1573943598
3	4	1732	great dialogue	1573943604
4	4	7569	so bad it's good	1573943455

In [105]:
```
tags['parsed_time'] = pd.to_datetime(tags['timestamp'], unit='s')
```

Data Type datetime64[ns] maps to either M8[ns] depending on the hardware

In [106]:
```
tags['parsed_time'].dtype
```

Out[106]:
```
dtype('<M8[ns]')
```

In [107]:
```
tags.head(2)
```

Out[107]:

	userId	movieId	tag	timestamp	parsed_time
0	3	260	classic	1439472355	2015-08-13 13:25:55
1	3	260	sci-fi	1439472256	2015-08-13 13:24:16

Selecting rows based on timestamps

In [108]:
```
greater_than_t = tags['parsed_time'] > '2015-02-01'

selected_rows = tags[greater_than_t]

tags.shape, selected_rows.shape
```

Out[108]:
```
((1093360, 5), (692112, 5))
```

Sorting the table using the timestamps

In [109]:
```
tags.sort_values(by='parsed_time', ascending=True)[:10]
```

Out[109]:

	userId	movieId	tag	timestamp	parsed_time
900600	129396	2788	monty python	1135429210	2005-12-24 13:00:10
900595	129396	1732	coen brothers	1135429236	2005-12-24 13:00:36
900592	129396	1206	stanley kubrick	1135429248	2005-12-24 13:00:48
900591	129396	1193	jack nicholson	1135429371	2005-12-24 13:02:51
900607	129396	5004	peter sellers	1135429399	2005-12-24 13:03:19
900589	129396	47	brad pitt	1135429412	2005-12-24 13:03:32
900590	129396	47	morgan freeman	1135429412	2005-12-24 13:03:32
900605	129396	4011	guy ritchie	1135429431	2005-12-24 13:03:51
900604	129396	4011	brad pitt	1135429431	2005-12-24 13:03:51
900588	129396	32	bruce willis	1135429442	2005-12-24 13:04:02

Average Movie Ratings over Time

Are Movie ratings related to the year of launch?

In [110]:
```
average_rating = ratings[['movieId','rating']].groupby('movieId', as_index=False).mean()
average_rating.tail()
```

Out[110]:

	movieId	rating
9061	161944	5.0
9062	162376	4.5
9063	162542	5.0
9064	162672	3.0
9065	163949	5.0

In [111]:
```
joined = movies.merge(average_rating, on='movieId', how='inner')
joined.head()
joined.corr()
```

Out[111]:

	movieId	rating
movieId	1.000000	-0.041213
rating	-0.041213	1.000000

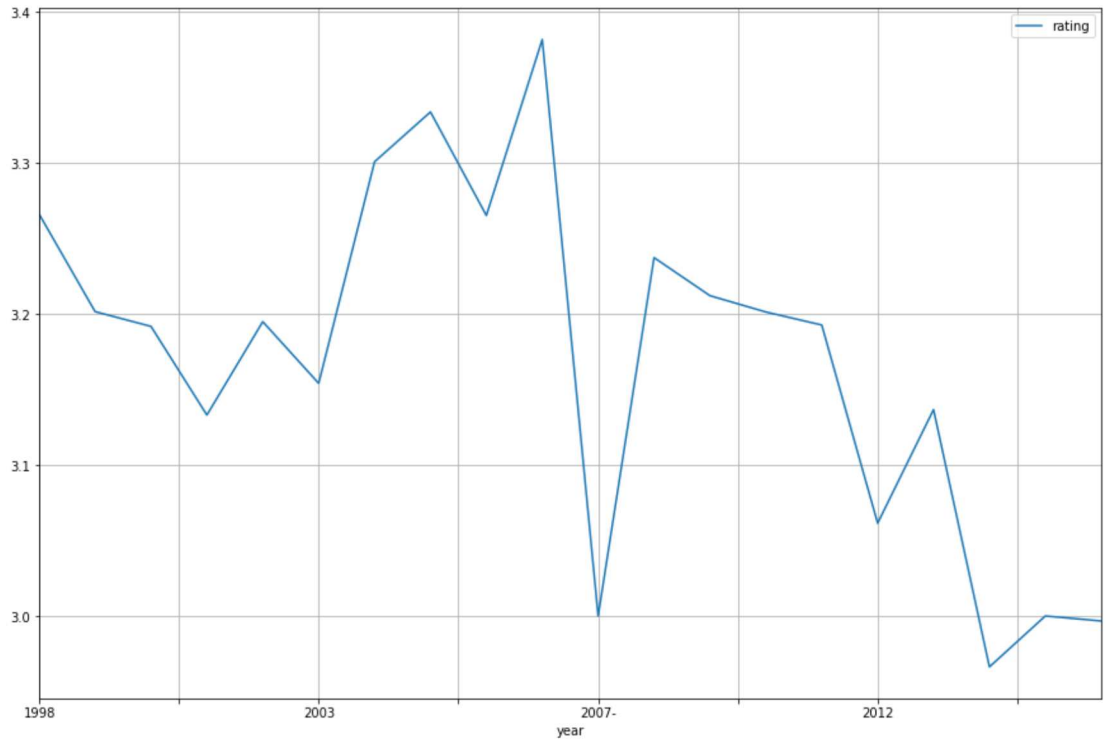
In [112]:
```
yearly_average = joined[['year','rating']].groupby('year', as_inde
x=False).mean()
yearly_average[:10]
```

Out[112]:

	year	rating
0	1902	4.333333
1	1915	3.000000
2	1916	3.500000
3	1917	4.250000
4	1918	4.250000
5	1919	3.000000
6	1920	2.500000
7	1921	4.387500
8	1922	3.926587
9	1923	4.166667

In [113]:
```
yearly_average[-20:].plot(x='year', y='rating', figsize=(15,10), gr
id=True)
```

Out[113]: <matplotlib.axes._subplots.AxesSubplot at 0x194a3c50>



Do some years look better for the boxoffice movies than others?

Does any data point seem like an outlier in some sense?

In [ ]:
```

```