02-A_Rate_Predictions

2017년 4월 13일

```
In [11]: # coding: utf-8

         import pandas as pd
         import numpy as np
         from matplotlib import rcParams
         import matplotlib.pyplot as plt
         from collections import defaultdict
         from datetime import datetime
         import matplotlib.patches as mpatches
         import matplotlib
         import time
         from __future__ import print_function
         %matplotlib inline

         rcParams['font.family'] = 'NanumGothic'
         rcParams.update({'font.size': 12})
         matplotlib.style.use('ggplot')
         pd.options.display.max_rows=14
```

0.1 무비 렌즈 데이터로 별점을 예측해 보자

- User Based 별점 예측
- Item(Movie) Based 별점 예측

Movie Lens 데이터 로드 http://grouplens.org/datasets/movielens/

```
In [12]: def movieLensDataLoad(type):
             ## user 영화 별점 data
```

```
    ratings = pd.read_csv("/Users/youngseoklee/Dropbox/fc-recsys-school-master/ch3/res

    ## movie meta(타이틀,장르) data
    movies = pd.read_csv("/Users/youngseoklee/Dropbox/fc-recsys-school-master/ch3/reso

    ## user가 영화에 tag를 기입한 data
    tags = pd.read_csv("/Users/youngseoklee/Dropbox/fc-recsys-school-master/ch3/resou
    # tags = pd.read_csv("/Users/goodvc/Documents/data-analytics/movie-recommendation/
    return ( ratings, movies, tags )

#ratings, movies, tags = movieLensDataLoad('ml-20m')
ratings, movies, tags = movieLensDataLoad('ml-latest-small')
```

In [13]:
```
#ratings = pd.read_csv("movieLens/ml-latest-small/ratings.csv")
ratings.head(3)
```

Out[13]:

	userId	movieId	rating	timestamp
0	1	31	2.5	1260759144
1	1	1029	3.0	1260759179
2	1	1061	3.0	1260759182

In [74]:
```
ratings
```

Out[74]:

	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
5	1	1263	2.0	1260759151
6	1	1287	2.0	1260759187
…	…	…	…	…
99997	671	5995	4.0	1066793014
99998	671	6212	2.5	1065149436
99999	671	6268	2.5	1065579370
100000	671	6269	4.0	1065149201
100001	671	6365	4.0	1070940363
100002	671	6385	2.5	1070979663

```
100003    671    6565    3.5  1074784724

[100004 rows x 4 columns]
```

In [96]:
```python
ratings.groupby(['userId'],).count()
```

Out[96]:
```
        movieId  rating  timestamp
userId
1            20      20         20
2            76      76         76
3            51      51         51
4           204     204        204
5           100     100        100
6            44      44         44
7            88      88         88
...         ...     ...        ...
665         434     434        434
666          40      40         40
667          68      68         68
668          20      20         20
669          37      37         37
670          31      31         31
671         115     115        115

[671 rows x 3 columns]
```

In [95]:
```python
ratings[ratings['userId']==6]
```

Out[95]:
```
     userId  movieId  rating   timestamp
451       6      111     4.0  1109258212
452       6      158     2.0  1108134263
453       6      173     2.0  1109258228
454       6      293     5.0  1108134539
455       6      596     4.0  1108134269
456       6      903     4.0  1108134299
457       6     1204     5.0  1108134266
..      ...      ...     ...         ...
488       6     7090     3.0  1108134534
```

```
489       6     7153     5.0  1108134519
490       6     7361     4.0  1108134524
491       6     8368     3.5  1108134526
492       6     8636     4.0  1108134537
493       6     8784     3.0  1108134531
494       6     8874     4.5  1108134521

[44 rows x 4 columns]
```

In [50]:
```
movies
```

Out[50]:
```
      movieId                                              title  \
0           1                                   Toy Story (1995)
1           2                                     Jumanji (1995)
2           3                            Grumpier Old Men (1995)
3           4                           Waiting to Exhale (1995)
4           5                 Father of the Bride Part II (1995)
5           6                                        Heat (1995)
6           7                                     Sabrina (1995)
…         …                                                  …
9118   162376                                   Stranger Things
9119   162542                                      Rustom (2016)
9120   162672                               Mohenjo Daro (2016)
9121   163056                               Shin Godzilla (2016)
9122   163949  The Beatles: Eight Days a Week - The Touring Y…
9123   164977                           The Gay Desperado (1936)
9124   164979                              Women of '69, Unboxed


                                           genres
0     Adventure|Animation|Children|Comedy|Fantasy
1                      Adventure|Children|Fantasy
2                                  Comedy|Romance
3                            Comedy|Drama|Romance
4                                          Comedy
5                           Action|Crime|Thriller
6                                  Comedy|Romance
…                                             …
```

```
9118                                          Drama
9119                              Romance|Thriller
9120                       Adventure|Drama|Romance
9121               Action|Adventure|Fantasy|Sci-Fi
9122                                    Documentary
9123                                         Comedy
9124                                    Documentary

[9125 rows x 3 columns]
```

In [108]:
```
movie_ratings = pd.merge(movies, ratings)
most_rated = movie_ratings.groupby('title').size().sort_values(ascending=False)[:25]
most_rated
```

Out[108]:
```
title
Forrest Gump (1994)                          341
Pulp Fiction (1994)                          324
Shawshank Redemption, The (1994)             311
Silence of the Lambs, The (1991)             304
Star Wars: Episode IV - A New Hope (1977)    291
Jurassic Park (1993)                         274
Matrix, The (1999)                           259
                                            ...
Aladdin (1992)                               215
Fugitive, The (1993)                         213
Dances with Wolves (1990)                    202
Fight Club (1999)                            202
Seven (a.k.a. Se7en) (1995)                  201
Usual Suspects, The (1995)                   201
Apollo 13 (1995)                             200
dtype: int64
```

In [114]:
```
movie_stats = movie_ratings.groupby('title').agg({'rating': [np.size, np.mean]})
movie_stats.head()
```

Out[114]:
```
                                         rating
                                           size      mean
title
```

"Great Performances" Cats (1998)	2.0	1.750000
$9.99 (2008)	3.0	3.833333
'Hellboy': The Seeds of Creation (2004)	1.0	2.000000
'Neath the Arizona Skies (1934)	1.0	0.500000
'Round Midnight (1986)	2.0	2.250000

In [115]:
```
# sort by rating average
movie_stats.sort_values([('rating', 'mean')], ascending=False).head()
```

Out[115]:

	rating	
	size	mean
title		
Ivan Vasilievich: Back to the Future (Ivan Vasi…	1.0	5.0
Alien Escape (1995)	1.0	5.0
Boiling Point (1993)	1.0	5.0
Bone Tomahawk (2015)	1.0	5.0
Borgman (2013)	1.0	5.0

In [116]:
```
atleast_100 = movie_stats['rating']['size'] >= 100
movie_stats[atleast_100].sort_values([('rating', 'mean')], ascending=False)[:15]
```

Out[116]:

	rating	
	size	mean
title		
Godfather, The (1972)	200.0	4.487500
Shawshank Redemption, The (1994)	311.0	4.487138
Godfather: Part II, The (1974)	135.0	4.385185
Usual Suspects, The (1995)	201.0	4.370647
Schindler's List (1993)	244.0	4.303279
One Flew Over the Cuckoo's Nest (1975)	144.0	4.256944
Fargo (1996)	224.0	4.256696
…	…	…
American Beauty (1999)	220.0	4.236364
Dark Knight, The (2008)	121.0	4.235537
Casablanca (1942)	117.0	4.235043
Star Wars: Episode V - The Empire Strikes Back …	234.0	4.232906
Memento (2000)	132.0	4.227273
Taxi Driver (1976)	118.0	4.224576

```
Monty Python and the Holy Grail (1975)             145.0  4.224138

[15 rows x 2 columns]
```

In [15]:
```
tags.head(2)
```

Out[15]:
```
   userId  movieId                       tag   timestamp
0      15      339  sandra 'boring' bullock  1138537770
1      15     1955                   dentist  1193435061
```

User Based 별점 예측

U(User) M(Movie)

1. U X M vector Matrix를 만든다. key가 userid, value가 { 'movieId':rating }
2. 나와 비슷한 유저를 찾는다.

In [51]:
```
## 1. U X M vector Matrix를 만든다.
UM_matrix_ds = ratings.pivot(index='userId', columns='movieId', values='rating')

print( "UM Matrix value size", UM_matrix_ds.values.size)
print( "ratings value size", ratings.values.size)
```

```
UM Matrix value size 6083286
ratings value size 400016
```

In [17]:
```
UM_matrix_ds.head(2)
```

Out[17]:
```
movieId  1       2       3       4       5       6       7       8       \
userId
1            NaN     NaN     NaN     NaN     NaN     NaN     NaN     NaN
2            NaN     NaN     NaN     NaN     NaN     NaN     NaN     NaN

movieId  9       10      ...     161084  161155  161594  161830  161918  \
userId                   ...
1            NaN     NaN ...        NaN     NaN     NaN     NaN     NaN
2            NaN     4.0 ...        NaN     NaN     NaN     NaN     NaN

movieId  161944  162376  162542  162672  163949
```

```
userId
1              NaN      NaN      NaN      NaN      NaN
2              NaN      NaN      NaN      NaN      NaN

[2 rows x 9066 columns]
```

In [119]:
```python
## 그럼 이제 최근접 이웃을 찾아 보자. 3가지 유사도 측정 함수를 이용할 수 있도록 정의함
## 유사하면 1, 다르면 0으로 수렴
import math
from operator import itemgetter
from scipy.spatial import distance

def distance_cosine(a,b):
    return 1-distance.cosine(a,b)

def distance_correlation(a,b):
    return 1-distance.correlation (a,b)

def distance_euclidean(a,b):
    return 1/(distance.euclidean(a,b)+1)
```

In [52]:
```python
## 유사도 측정 함수
def nearest_neighbor_user( user, topN, simFunc ) :
    u1 = UM_matrix_ds.loc[user].dropna()
    ratedIndex = u1.index
    nn = {}

    ## Brute Force Compute
    for uid, row in UM_matrix_ds.iterrows():
        interSectionU1 = []
        interSectionU2 = []
        if uid==user:
            continue

        for i in ratedIndex:
            if False==math.isnan(row[i]):
                interSectionU1.append(u1[i])
```

```
                    interSectionU2.append(row[i])
                interSectionLen = len(interSectionU1)


                ## At least 3 intersection items
                if interSectionLen < 3 :
                    continue

                ## similarity functon
                sim = simFunc(interSectionU1,interSectionU2)

                if  math.isnan(sim) == False:
                    nn[uid] = sim

            ## top N returned
            return sorted(nn.items(),key=itemgetter(1))[:-(topN+1):-1]
```

In [121]:
```
st=time.time()
print(nearest_neighbor_user(6, 50, distance_euclidean))
#print(nearest_neighbor_user(6, 50, distance_cosine))
#print(nearest_neighbor_user(6, 50, distance_correlation))

print(time.time()-st, 'sec')
```

```
[(81, 1.0), (661, 0.6666666666666666), (434, 0.585786437626905), (40, 0.585786437626905), (455
0.498565912247 sec
```

In [125]:
```
def predictRating(userid, nn=50, simFunc=distance_euclidean) :

    ## neighboorhood
    neighbor = nearest_neighbor_user(userid,nn,simFunc)
    neighbor_id = [id for id,sim in neighbor]

    ## neighboorhood's movie : al least 4 ratings
    neighbor_movie = UM_matrix_ds.loc[neighbor_id]\
                     .dropna(1, how='all', thresh = 4 )
    #neighbor_movie.head()
```

```
        neighbor_dic = (dict(neighbor))
        ret = [] # ['movieId', 'predictedRate']

        ## rating predict by my similarities
        for movieId, row in neighbor_movie.iteritems():
            jsum, wsum = 0, 0
            for v in row.dropna().iteritems():
                sim = neighbor_dic.get(v[0],0)
                jsum += sim
                wsum += (v[1]*sim)
            ret.append([movieId, wsum/jsum])

        return ret
```

In [126]:
```
(predictRating(6, 50))
```

Out[126]:
```
[[1, 3.9649721588824884],
 [2, 3.6819288235729339],
 [5, 2.7140065408290037],
 [10, 3.8739509361270414],
 [11, 3.6899419687602002],
 [16, 3.6406088788359048],
 [17, 4.1277118490649523],
 [19, 3.1917954600830631],
 [21, 3.5559058576381273],
 [22, 3.3858378101544711],
 [24, 3.4493068220757404],
 [25, 4.0531498077989214],
 [29, 4.1931528550856862],
 [31, 3.3893792999440309],
 [32, 3.8183131810518196],
 [34, 3.5424517263378914],
 [36, 4.1059036276478276],
 [39, 3.6614065626461243],
 [44, 2.9252327037256158],
 [45, 4.0],
 [47, 4.0753734050339094],
```

```
[48, 2.702523184348669],
[50, 4.3700795810876381],
[52, 4.1055502036897318],
[58, 3.747222958200171],
[104, 3.5621611863589737],
[110, 3.7053158490453395],
[111, 4.3330633698587215],
[141, 3.2842056395730372],
[145, 3.6098774296597167],
[147, 3.8560296461482806],
[150, 3.8242135325428253],
[153, 2.710239695862088],
[156, 3.3318474932286817],
[158, 2.565624523201786],
[160, 3.0181860369898774],
[161, 4.0],
[162, 4.3433776985682417],
[163, 2.9147764484239511],
[165, 3.3724756276574746],
[168, 3.0473967110973872],
[172, 2.582647187206351],
[173, 2.9529431396511523],
[185, 3.0335062698005473],
[194, 4.1773966907386777],
[196, 3.3255708284000991],
[198, 3.7004949565777387],
[207, 3.5318346587246228],
[208, 2.7519224060665586],
[216, 2.8066185893906477],
[223, 3.9579591050505702],
[227, 2.9952820140376359],
[230, 3.5753164962433317],
[231, 3.0109707732193987],
[232, 4.2656505858627147],
[235, 4.0],
[236, 2.3340106400256899],
```

```
[237, 3.5308803845224306],
[246, 4.3283007905656463],
[252, 3.0],
[253, 3.4427918258415171],
[260, 3.7996812683392549],
[265, 4.3281168696796311],
[266, 4.0453733348207752],
[272, 3.7826275177455178],
[273, 3.0318643684734559],
[277, 3.4807362591321942],
[288, 3.6874575989641407],
[292, 3.2621770212165808],
[293, 4.2814039568388162],
[296, 4.4225091119474369],
[299, 3.225911993741573],
[300, 3.7747207230735644],
[316, 3.6335304675716249],
[317, 3.135944069562298],
[318, 4.1789710972183887],
[319, 4.3580321490076335],
[327, 2.6496998448674325],
[329, 3.3131794862499753],
[333, 3.1661774555275906],
[337, 4.2874039118457832],
[339, 3.3115284626763617],
[342, 3.898230951147124],
[344, 3.2358984179065122],
[348, 3.9844680937385437],
[349, 4.0645526543398196],
[350, 3.6480688072461249],
[353, 3.167432038500404],
[355, 3.0000000000000004],
[356, 3.8453810912181079],
[357, 3.8140187371542513],
[362, 3.7097729461168463],
[364, 3.8787976053562865],
```

```
[367, 3.1189827186362358],
[368, 3.8124655052894694],
[370, 2.3577251034933502],
[372, 2.3386716572371493],
[374, 2.1617953540857044],
[377, 3.8102398421884875],
[380, 3.6733010220935292],
[410, 2.8847272012882672],
[412, 3.1268790920463387],
[420, 2.8616330925799147],
[432, 2.7737175923002253],
[434, 2.9999999999999996],
[435, 2.1045883547148341],
[440, 3.5862872862650126],
[442, 3.1473902987214042],
[454, 3.4805344841905002],
[455, 3.2962322634642662],
[457, 4.1209217382599963],
[466, 2.7565461765549686],
[471, 4.0273475420998528],
[474, 4.2805029147903451],
[475, 4.5773502691896262],
[480, 3.7834797190946889],
[481, 3.618739181186569],
[485, 2.9883863902275962],
[489, 2.7990760591136432],
[494, 3.3208891956450231],
[497, 4.1164769059482733],
[500, 3.9201551776061256],
[508, 3.5495059829643796],
[509, 2.9760697625618326],
[515, 4.4759700849721558],
[519, 2.2013425146491614],
[520, 3.3090323851795738],
[527, 3.9651448792745803],
[538, 4.2416145476651526],
```

```
[539, 3.3032067383794459],
[541, 4.0833646426150922],
[542, 2.5308803845224306],
[543, 3.3151811446336832],
[551, 3.0666060580747221],
[553, 3.5911172053606935],
[555, 3.7053592392438195],
[585, 3.2255613524822215],
[586, 3.4386469149325722],
[587, 3.6136218141023204],
[588, 3.8159176678510471],
[589, 3.7736905429841556],
[590, 3.5202837251356249],
[592, 3.506762156585836],
[593, 3.873559376275781],
[594, 3.6559408257248975],
[595, 3.8447551698975868],
[596, 3.93174978160559],
[597, 3.5245172670188527],
[608, 4.4651192092621956],
[616, 3.7804174163413551],
[628, 3.2294965313519644],
[648, 4.2048648181368886],
[653, 3.4965565286690019],
[661, 3.8044351068109039],
[724, 3.3997854186006791],
[733, 3.8752093323038284],
[750, 4.5730713549056494],
[778, 4.1480716287512207],
[780, 3.403124719499715],
[785, 2.7415833859624605],
[858, 4.395282937141598],
[899, 3.9374005800574503],
[902, 4.0336210454983021],
[903, 4.270423692692507],
[904, 4.320640928072387],
```

[908, 4.3677242256158779],
[910, 4.577115852658002],
[912, 3.8927554674670644],
[913, 4.5093901556397409],
[914, 3.3282125905598337],
[919, 3.9061712295698703],
[923, 4.3660244766925453],
[924, 3.5744035229481947],
[926, 4.7391143198029111],
[953, 3.7707691592774268],
[1019, 3.5036064384794483],
[1022, 3.7759907622602045],
[1027, 2.896725033015541],
[1028, 3.3070259028233853],
[1029, 4.0144763912269417],
[1035, 4.5299065591196799],
[1036, 3.9076949544803634],
[1073, 3.7605594710157333],
[1077, 4.3803831785549425],
[1079, 4.2974724638174235],
[1080, 4.2140964337003401],
[1081, 3.7247832105961192],
[1084, 4.1651740834003093],
[1089, 4.2146123886186162],
[1094, 3.8446950300056417],
[1096, 4.3452361501004564],
[1097, 3.7248868882374491],
[1103, 3.9293167013040478],
[1104, 4.3720973734755111],
[1124, 3.81824655451104],
[1125, 3.4551442830049388],
[1132, 3.3123574198023178],
[1136, 4.1171604385483214],
[1172, 5.0],
[1185, 4.3555736770021385],
[1188, 3.785263600831537],

```
[1193, 4.3060131824292665],
[1196, 4.1850254029529363],
[1197, 3.8603430615749743],
[1198, 3.9635558143072691],
[1199, 4.3831780861536931],
[1200, 3.3974208578936254],
[1201, 3.900100119891857],
[1203, 4.6743837026427082],
[1206, 3.9377790538806487],
[1207, 3.8831473666840801],
[1208, 4.2505004647663069],
[1210, 3.4774127436594995],
[1213, 4.00049524383678],
[1214, 3.4433768778146709],
[1219, 4.3788504389188514],
[1220, 3.927407882636488],
[1221, 4.1921750696825422],
[1222, 4.1238686348927249],
[1225, 4.6722649944594927],
[1228, 4.5244100775258334],
[1230, 4.4768842800629027],
[1231, 3.8864125840390171],
[1235, 4.1006053906388678],
[1240, 4.1121891616262776],
[1242, 3.6311299163077373],
[1244, 4.0942666347644145],
[1246, 4.4981778310020566],
[1247, 4.5500654252133756],
[1250, 4.2557923385688428],
[1252, 4.393298058570541],
[1258, 4.134935966939234],
[1259, 4.0032964663002755],
[1263, 4.5377150530178874],
[1265, 3.8879886182686798],
[1270, 3.9297124259222938],
[1278, 3.8373742680485559],
```

```
[1282, 3.4843954782043993],
[1285, 3.9384228241916377],
[1288, 4.4423745723606016],
[1291, 3.9739879042624615],
[1292, 4.1918916079923685],
[1293, 3.1273467592036348],
[1300, 4.0505032084901469],
[1302, 3.7837402419071444],
[1304, 4.4243157537787789],
[1307, 3.9740870055162283],
[1333, 3.9896856313446132],
[1372, 3.6915505987617534],
[1374, 3.7565146673930903],
[1376, 3.0651218317630753],
[1380, 3.1839252696007807],
[1387, 3.9873187523991622],
[1393, 3.8658821212299506],
[1394, 3.7722101922950819],
[1513, 4.0969109834557234],
[1544, 3.135631687202491],
[1580, 3.6171946486201718],
[1584, 3.5320916583739246],
[1617, 3.8001396453116136],
[1639, 2.5791139714575113],
[1653, 3.8795656220395407],
[1674, 4.2430288887812182],
[1682, 3.7629336288657815],
[1704, 4.5668065991639084],
[1721, 3.1473186038591519],
[1732, 3.561431724710292],
[1748, 3.4246048239619387],
[1784, 3.8610304422793367],
[1923, 4.2991910833871216],
[1958, 3.7912568721432356],
[1961, 4.0],
[1968, 3.6368252742651852],
```

```
[2010, 4.3649062839145678],
[2011, 3.2134348191181048],
[2019, 4.8074188703705589],
[2020, 3.2774281270360177],
[2028, 3.6712224195744279],
[2054, 2.7447511197474199],
[2081, 3.9437509504187358],
[2100, 3.1180109702408845],
[2134, 2.978919283009501],
[2144, 3.2357002877169161],
[2150, 3.6063581988507645],
[2174, 3.4922313945589711],
[2248, 3.9519941660204658],
[2268, 3.0991360800144241],
[2294, 3.7242646479029924],
[2302, 4.1878339662915893],
[2321, 3.4472685718449374],
[2324, 4.0695914520265832],
[2329, 4.3592989900991368],
[2352, 3.7933732649368124],
[2355, 3.6250915096977163],
[2390, 3.637219799461072],
[2395, 3.9591122298713324],
[2396, 3.8772567922383083],
[2406, 3.1338724422205746],
[2469, 3.623181204849391],
[2502, 3.8179905528639515],
[2539, 3.2375453211467797],
[2599, 4.3984123784022779],
[2683, 2.8694219382854609],
[2690, 4.2631471428658756],
[2692, 3.7824440391470531],
[2700, 3.6481774122501722],
[2706, 4.0],
[2712, 4.4378696795686388],
[2716, 3.6852440247716021],
```

```
[2762, 4.3427784796187643],
[2770, 3.3946927983076529],
[2791, 3.5348654981079948],
[2797, 3.5680448939105385],
[2858, 4.138491371732135],
[2918, 3.8790720166334878],
[2959, 3.9772728258054553],
[2987, 3.9120299099522402],
[2997, 4.0743810650478585],
[3114, 4.092658972065081],
[3160, 3.4078193342094525],
[3176, 3.7795514629736382],
[3253, 3.1732882869568497],
[3408, 3.9194616919692571],
[3481, 3.3301075092434731],
[3535, 3.2334108795367911],
[3578, 3.5908894240421656],
[3623, 3.0697570975966548],
[3624, 3.5300541541001791],
[3753, 3.0763626631969627],
[3755, 2.9789472860326498],
[3793, 3.0006431357248413],
[3897, 3.9126706873509067],
[3948, 3.3412192710625566],
[3977, 3.3822989209028465],
[4226, 4.7963639234248969],
[4246, 3.0953878768950833],
[4306, 3.3517230463334045],
[4878, 3.3474296571736848],
[4886, 3.4617298231060816],
[4896, 4.0909423669028318],
[4963, 2.940214107185108],
[4973, 4.1526160521681303],
[4993, 4.6579061082365447],
[5060, 4.1542665504381606],
[5299, 3.5318971894858229],
```

```
 [5349, 3.5623584113201043],
 [5445, 4.2232626302033891],
 [5618, 4.2441228540935017],
 [5669, 4.1849189024046884],
 [5816, 3.6083216722535529],
 [5952, 4.6448564285521163],
 [5995, 3.5416787153238238],
 [6333, 3.6772963876738629],
 [6377, 3.5979508253896135],
 [6539, 3.3855886722599147],
 [6874, 3.3044663720020888],
 [7153, 4.5598312562059453],
 [7361, 4.1147560010400737],
 [8636, 3.9143066739315984],
 [8874, 4.109229801563524],
 [8961, 3.7298022694076067],
 [30749, 3.7616949117039491],
 [46578, 4.4854860607825495],
 [48394, 3.8713263989845137],
 [48516, 4.2606414045099745],
 [56367, 3.6633322065872878],
 [58559, 2.9849587020571238],
 [68157, 4.3203727088755643],
 [70286, 4.1974550690481429],
 [79132, 3.7074156720249265],
 [122886, 3.8970725856416633]]
```

In [99]:
```
## user의 별점 매긴 영화와 영화 정보 높은 별점순으로 보기
def ratingMovies(userid):
    ds = pd.merge(ratings[ratings.userId==userid], movies, on=['movieId'])
    return ds.sort_values(['rating'],ascending=False)[['rating','title','genres','movi
ratingMovies(6).head(20)
```

Out[99]:
```
    rating                                             title  \
3      5.0  Léon: The Professional (a.k.a. The Professiona…
36     5.0      Lord of the Rings: The Two Towers, The (2002)
38     5.0  Lord of the Rings: The Return of the King, The…
```

```
6       5.0                  Lawrence of Arabia (1962)
43      4.5                  Shaun of the Dead (2004)
8       4.5                        Stand by Me (1986)
28      4.5                    Iron Giant, The (1999)
..      ...                                       ...
41      4.0                      Spider-Man 2 (2004)
26      4.0          Run Lola Run (Lola rennt) (1998)
23      4.0               Planet of the Apes (1968)
20      4.0                        Beetlejuice (1988)
19      4.0                        'burbs, The (1989)
0       4.0                        Taxi Driver (1976)
4       4.0                          Pinocchio (1940)

                                      genres  movieId
3               Action|Crime|Drama|Thriller      293
36                        Adventure|Fantasy     5952
38         Action|Adventure|Drama|Fantasy     7153
6                      Adventure|Drama|War     1204
43                           Comedy|Horror     8874
8                          Adventure|Drama     1259
28  Adventure|Animation|Children|Drama|Sci-Fi     2761
..                                       ...      ...
41            Action|Adventure|Sci-Fi|IMAX     8636
26                            Action|Crime     2692
23                     Action|Drama|Sci-Fi     2529
20                          Comedy|Fantasy     2174
19                                  Comedy     2072
0                     Crime|Drama|Thriller      111
4       Animation|Children|Fantasy|Musical      596

[20 rows x 4 columns]
```

```
In [100]: def join_movie_info( predicted_result ):
              predicted_ratings = pd.DataFrame(predicted_result, columns=['movieId', 'predicte
              result_ds = pd.merge( movies[movies.movieId > 0], predicted_ratings, on=['movieId
              return result_ds.sort_values(['predicted_rating'], ascending=False)
```

```python
result = predictRating(6);
join_movie_info(result)
```

Out[100]:

```
     movieId                                              title  \
198     1172     Cinema Paradiso (Nuovo cinema Paradiso) (1989)
275     2019         Seven Samurai (Shichinin no samurai) (1954)
334     4226                                     Memento (2000)
173      926                                All About Eve (1950)
208     1203                                 12 Angry Men (1957)
219     1225                                     Amadeus (1984)
342     4993   Lord of the Rings: The Fellowship of the Ring,…
..       …                                                  …
131      542                                 Son in Law (1993)
95       370           Naked Gun 33 1/3: The Final Insult (1994)
96       372                              Reality Bites (1994)
56       236                                French Kiss (1995)
125      519                                  RoboCop 3 (1993)
97       374                                Richie Rich (1994)
105      435                                  Coneheads (1993)

                                   genres  predicted_rating
198                                 Drama          5.000000
275                Action|Adventure|Drama          4.807419
334                      Mystery|Thriller          4.796364
173                                 Drama          4.739114
208                                 Drama          4.674384
219                                 Drama          4.672265
342                     Adventure|Fantasy          4.657906
..                                     …                 …
131                  Comedy|Drama|Romance          2.530880
95                          Action|Comedy          2.357725
96                   Comedy|Drama|Romance          2.338672
56                  Action|Comedy|Romance          2.334011
125  Action|Crime|Drama|Sci-Fi|Thriller          2.201343
97                        Children|Comedy          2.161795
```

```
105                         Comedy|Sci-Fi          2.104588

[371 rows x 4 columns]
```

In [101]:
```
## 6번 유저의 별점 예측
userid=6
pd.merge(ratingMovies(userid), join_movie_info(predictRating(userid)), on=['movieId'
    .sort_values(['predicted_rating'], ascending =False)\
```

Out[101]:
```
     rating                                   title_x  \
20      NaN                                       NaN
7       4.0  Seven Samurai (Shichinin no samurai) (1954)
21      NaN                                       NaN
22      NaN                                       NaN
23      NaN                                       NaN
24      NaN                                       NaN
25      NaN                                       NaN
..      ...                                       ...
364     NaN                                       NaN
365     NaN                                       NaN
366     NaN                                       NaN
367     NaN                                       NaN
368     NaN                                       NaN
369     NaN                                       NaN
370     NaN                                       NaN

                   genres_x  movieId  \
20                      NaN     1172
7    Action|Adventure|Drama     2019
21                      NaN     4226
22                      NaN      926
23                      NaN     1203
24                      NaN     1225
25                      NaN     4993
..                      ...      ...
364                     NaN      542
365                     NaN      370
```

```
366                     NaN      372
367                     NaN      236
368                     NaN      519
369                     NaN      374
370                     NaN      435

                                                title_y  \
20     Cinema Paradiso (Nuovo cinema Paradiso) (1989)
7         Seven Samurai (Shichinin no samurai) (1954)
21                                     Memento (2000)
22                               All About Eve (1950)
23                                12 Angry Men (1957)
24                                    Amadeus (1984)
25   Lord of the Rings: The Fellowship of the Ring,…
..                                                …
364                                 Son in Law (1993)
365         Naked Gun 33 1/3: The Final Insult (1994)
366                              Reality Bites (1994)
367                                French Kiss (1995)
368                                 RoboCop 3 (1993)
369                                Richie Rich (1994)
370                                  Coneheads (1993)

                       genres_y  predicted_rating
20                        Drama          5.000000
7        Action|Adventure|Drama          4.807419
21             Mystery|Thriller          4.796364
22                        Drama          4.739114
23                        Drama          4.674384
24                        Drama          4.672265
25            Adventure|Fantasy          4.657906
..                          …               …
364        Comedy|Drama|Romance          2.530880
365               Action|Comedy          2.357725
366        Comedy|Drama|Romance          2.338672
367       Action|Comedy|Romance          2.334011
```

```
368  Action|Crime|Drama|Sci-Fi|Thriller         2.201343
369                       Children|Comedy         2.161795
370                         Comedy|Sci-Fi         2.104588

[371 rows x 7 columns]
```

In [102]:
```
eval_ratings = ratings
```

In [103]:
```
# ratings['userId'].drop_duplicates().values[:]
def eval_prediction( predict_users,  n_users=50 ):
    ## evaluation
    ds = pd.merge(eval_ratings,
                  ratings[['movieId','rating']].groupby(['movieId']).mean().rese
                  on='movieId', how='left')

    ds = ds.rename(columns= {'rating_x':'rating', 'rating_y':'mean_rating'})

    st = time.time()
    ## udpate to predict_rating
    distance_functions = [ ('euclidean',distance_euclidean), ('cosine', distance_cos
    for name, func in distance_functions:
        ds[name] = 0
        for userId in predict_users:
            for x in predictRating(userId, n_users, func):
                ds.loc[(ds.userId==userId) & (ds.movieId==x[0]),name]=x[1]
    print('elapsed', round(time.time()-st,2), 'sec')
    return ds[ds.euclidean+ds.cosine>0]
```

In [104]:
```
## 전체 userId list
users = UM_matrix_ds.index.tolist()
```

In [127]:
```
users[:5]
```

Out[127]:
```
[1, 2, 3, 4, 5]
```

In [129]:
```
## 1,2, 3, ... n 명 별점 예측, 시간은 얼마나 걸릴까?
predicted = eval_prediction(users[:1], 100 )
predicted = eval_prediction(users[:2], 100 )
```

```
predicted = eval_prediction(users[:3], 100 )
predicted = eval_prediction(users[:10], 100 )
```

```
elapsed 13.88 sec
elapsed 19.78 sec
elapsed 23.67 sec
elapsed 55.51 sec
```

In [130]:
```
predicted
```

Out[130]:

	userId	movieId	rating	timestamp	mean_rating	euclidean	cosine
0	1	31	2.5	1260759144	3.178571	2.877513	2.943252
1	1	1029	3.0	1260759179	3.702381	3.445023	3.720016
2	1	1061	3.0	1260759182	3.545455	3.487024	3.500408
3	1	1129	2.0	1260759185	3.312500	3.095070	3.253296
4	1	1172	4.0	1260759205	4.260870	4.188129	4.095786
5	1	1263	2.0	1260759151	3.864583	3.572964	3.917519
6	1	1287	2.0	1260759187	3.891304	3.812625	3.941325
..	...	...	...	...	...	...	...
763	10	1358	5.0	942766420	3.886364	4.828427	4.667564
768	10	1690	3.0	942766679	3.033333	0.000000	3.798672
769	10	1704	4.0	942766472	4.140127	4.251785	4.520997
772	10	1923	5.0	942766515	3.552846	0.000000	4.416727
775	10	2406	4.0	942767328	3.584615	0.000000	4.125248
778	10	2571	5.0	942766515	4.183398	4.473794	4.639886
781	10	2840	3.0	942766213	2.750000	3.216944	0.000000

[485 rows x 7 columns]

In [131]:
```
predicted = predicted[ (predicted['cosine'] > 0) & (predicted['euclidean'] > 0) ]
```

In [132]:
```
def RMSE(X, left_col, right_col):
    return(np.sqrt(np.mean( (X[left_col] - X[right_col])**2 )))

def MAE(X, left_col, right_col):
    return(np.mean(np.absolute(X[left_col] - X[right_col])) )
```

In [133]:
```
MAE( predicted, 'rating', 'cosine')
```

```
Out[133]: 0.5543758106533989

In [134]: MAE( predicted, 'rating', 'euclidean')

Out[134]: 0.45777526449458195

In [135]: MAE( predicted, 'rating', 'mean_rating')

Out[135]: 0.6467212196321311

In [136]: for name in ['mean_rating', 'cosine', 'euclidean']:
              print ("MAE of {0} is {1} ".format(name, MAE( predicted, 'rating', name )))

          for name in ['mean_rating', 'cosine', 'euclidean']:
              print ("RMSE of {0} is {1} ".format(name, RMSE( predicted, 'rating', name )))

MAE of mean_rating is 0.646721219632
MAE of cosine is 0.554375810653
MAE of euclidean is 0.457775264495
RMSE of mean_rating is 0.814675719677
RMSE of cosine is 0.694610235557
RMSE of euclidean is 0.606544084213

In [137]: predicted = eval_prediction(users[:2], 20 )

elapsed 4.63 sec

In [ ]:
```