# Assignment #6

Wednesday, March 21TH, 2018

RANDY DO

# Contents

# Problem 1

Find 3 users who are closest to you in terms of age, gender, and occupation. For each of those 3 users:

- what are their top 3 favorite films?
- bottom 3 least favorite films?

Based on the movie values in those 6 tables (3 users X (favorite + least)), choose a user that you feel is most like you. Feel free to note any outliers (e.g., "I mostly identify with user 123, except I did not like ``Ghost'' at all").

This user is the "substitute you".

Program I will use to find this: closestuser.py
The if statement finds age of 24, Male, and student. Afterwards, it writes it in the text file 'clostest3users.txt'

```
users = []
fh_output = open('closest3users.txt', 'w')

with open('u.txt') as fp_input:
    for line in fp_input:
        users.append(line.split('|'))

for user in users:
    if (user[1] < '25' and user[1] > '23' and user[2] == 'M' and user[3] == 'student'):
        fh_output.write(str(user[0])+'|')
        fh_output.write(str(user[1])+'|')
        fh_output.write(str(user[2])+'|')
        fh_output.write(str(user[3])+'|')
        fh_output.write(str(user[4]))

fp_input.close()
fh_output.close()
```

I got 6 different results. I am going to choose the first three on the list. (

```
73|24|M|student|41850
301|24|M|student|55439
369|24|M|student|91335
472|24|M|student|87544
517|24|M|student|55454
641|24|M|student|60626
```

```
73|24|M|student|41850
301|24|M|student|55439
369|24|M|student|91335
```

Next, we find the top 3 and bottom 3.
The python file I will be using to find the top 3 and bottom 3
Topbottom.py
The inputs I am using are:
closest3users.txt – Got from closestuser.py
u.data – Gives id and rating of the users
u.item – Gives data of films

Since, I am fetching data from other files, I will be using itemgetter from operator. The output is topbottom.txt

```
import sys
from operator import itemgetter

users = []
ratings = []
films = []
fh_output = open('topbottom.txt', 'w')

with open('closest3users.txt') as users_input:
        for line in users_input:
                users.append(line.split('|'))

with open('u.data') as ratings_input:
        for line in ratings_input:
                ratings.append(line.split('\t'))

with open('u.item') as films_input:
        for line in films_input:
                films.append(line.split('|'))
first_user_ratings = []
second_user_ratings = []
third_user_ratings = []
```

I will do both top and bottom at the same time with this python file.

This will setup the rating for the 3 users by using itemgetter

```
i = 0
for user in users:
    i = i+1
    for rating in ratings:
        if (user[0] == rating[0]):
            if (i==1):
                first_user_ratings.append(rating)
            if (i==2):
                second_user_ratings.append(rating)
            if (i==3):
                third_user_ratings.append(rating)

first_user_ratings = sorted(first_user_ratings , key=itemgetter(2))
second_user_ratings = sorted(second_user_ratings , key=itemgetter(2))
third_user_ratings = sorted(third_user_ratings , key=itemgetter(2))
```

This will find the first user rating, top movie and bottom movie:

```
# First user top and bottom films

first_user_bottom = []
first_user_top = []

for i in range(0,3):
    first_user_bottom.append(first_user_ratings[i])

for i in range(len(first_user_ratings)-3, len(first_user_ratings)):
    first_user_top.append(first_user_ratings[i])

film_first_bottom = []
for rat in first_user_bottom:
    film_first_bottom.append(rat[1])

film_first_top = []
for rat in first_user_top:
        film_first_top.append(rat[1])

fh_output.write('First User id: ' + first_user_ratings[0][0])
fh_output.write('\n--------------\n')

fh_output.write('Top 3 films:')
fh_output.write('\n--------------\n')

for film in film_first_top:
    for movie in films:
        if (film == movie[0]):
            fh_output.write(movie[1] + '\n')

fh_output.write('-----------------------------------------\n')
fh_output.write('Bottom 3 films:')
fh_output.write('\n----------------\n')

for film in film_first_bottom:
        for movie in films:
                if (film == movie[0]):
                        fh_output.write(movie[1] + '\n')

fh_output.write('-----------------------------------------\n')
```

Result of the first user:

```
First User id: 73
--------------
Top 3 films:
--------------
Godfather: Part II, The (1974)
Graduate, The (1967)
Full Metal Jacket (1987)
-------------------------------------------
Bottom 3 films:
-----------------
Home Alone (1990)
Home Alone 3 (1997)
Beauty and the Beast (1991)
-------------------------------------------
```

To find the other user: I just change the variable first_user_rating->
second_user_ratings
third_user_ratings
Final results:

```
First User id: 73
--------------
Top 3 films:
--------------
Godfather: Part II, The (1974)
Graduate, The (1967)
Full Metal Jacket (1987)
-------------------------------------------
Bottom 3 films:
-----------------
Home Alone (1990)
Home Alone 3 (1997)
Beauty and the Beast (1991)
-------------------------------------------
Second User id: 301
--------------
Top 3 films:
--------------
Rock, The (1996)
Die Hard 2 (1990)
Fargo (1996)
-------------------------------------------
Bottom 3 films:
-----------------
Robin Hood: Men in Tights (1993)
Natural Born Killers (1994)
Nutty Professor, The (1996)
-------------------------------------------
Third User id: 369
--------------
Top 3 films:
--------------
Chasing Amy (1997)
As Good As It Gets (1997)
Dead Poets Society (1989)
-------------------------------------------
Bottom 3 films:
-----------------
Booty Call (1997)
How to Be a Player (1997)
Beautician and the Beast, The (1997)
-------------------------------------------
```

# Problem 2

Which 5 users are most correlated to the substitute you? Which 5 users are least correlated (i.e., negative correlation)?

From the list. I will be using ID: 73 as my substitute.
I did some google searching and how a similar program that'll help me in this assignment. It is from Programming-Collective-Intelligence Chapter 2 "recommendations.py"

*I added modified the TopMatches function in recommendation.py
When I call this function, I can add in my ID to filter out. As X = 73, it will search/identified my ID

```
def topMatches(x, prefs, person, n=5, similarity=sim_pearson):

    scores = [(similarity(prefs, person, other), other) for other in prefs
              if other != person]
    scores.sort()
    if x:
        scores.reverse()
    return scores[0:n]
```

I will use it in my correlated.py program

```
import recommendations as rec

pref= rec.loadMovieLens()
correlated = rec.topMatches(1, pref, '73')
noncorrelated =  rec.topMatches(0, pref, '73')
print "Five Most Correlated Users: " + '73'
print "------------------------------------------------------------"
for user in correlated:
    print user[1]

print "------------------------------------------------------------"

print "Five Least Correlated Users: " + '73'
print "------------------------------------------------------------"
for user in noncorrelated:
    print user[1]
```

```
Five Most Correlated Users: 73
------------------------------------------------------------
879
353
732
636
571
------------------------------------------------------------
Five Least Correlated Users: 73
------------------------------------------------------------
837
941
779
134
300
```

# Problem 3

Compute ratings for all the films that the substitute you have not seen. Provide a list of the top 5 recommendations for films that the substitute you should see. Provide a list of the bottom 5 recommendations (i.e., films the substitute you is almost certain to hate).

From the list. I will be using ID: 73 as my substitute. It is very similar to the last problem. Luckily, there is a function that does the recommendation!

The program that I will be using is: top5recommendation.py

```
import recommendations as rec

pref= rec.loadMovieLens()
recom = rec.getRecommendations(pref, '73')

print "Five Most recommended Movies for My Substitute: " + '73'
print "------------------------------------------------------------"
for movie in recom[:5]:
        print movie[1]

print "------------------------------------------------------------"

print "Five Least Recommended Movies for My Substitute: " + '73'
print "------------------------------------------------------------"
for movie in recom[-5:]:
        print movie[1]

print "------------------------------------------------------------"
```

```
Five Most recommended Movies for My Substitute: 73
-----------------------------------------------------
Tough and Deadly (1995)
They Made Me a Criminal (1939)
Star Kid (1997)
Someone Else's America (1995)
Santa with Muscles (1996)
-----------------------------------------------------
Five Least Recommended Movies for My Substitute: 73
-----------------------------------------------------
Amityville: A New Generation (1993)
Amityville Curse, The (1990)
Amityville 3-D (1983)
Amityville 1992: It's About Time (1992)
3 Ninjas: High Noon At Mega Mountain (1998)
-----------------------------------------------------
```

# Problem 4

Choose your (the real you, not the substitute you) favorite and least favorite film from the data. For each film, generate a list of the top 5 most correlated and bottom 5 least correlated films. Based on your knowledge of the resulting films, do you agree with the results? In other words, do you personally like / dislike the resulting films?

For this problem, again, I will be using the recommendation.py and get a movie of my favorite and a movie of my least favorite. The movies I will use are:

Favorite: Toy Story (1995)
Least Favorite: Braveheart (1995)

The program that I will be using is: myrecommendation.py

```
import recommendations as rec

pref= rec.loadMovieLens()

top = 'Toy Story (1995)'
bot = 'Braveheart (1995)'

topmov = rec.calculateSimilarItems(1, pref, 5)
botmov = rec.calculateSimilarItems(0, pref, 5)

print 'Best Recommended movies for ' + top + ':'
print '-----------------------------------------------------------------'
for movie in topmov[top]:
    print movie[1]
print "-----------------------------------------------------------"
print 'Least Recommended movies for ' + top + ':'
print '-----------------------------------------------------------------'
for movie in botmov[top]:
    print movie[1]
print "-----------------------------------------------------------"
print 'Best Recommended movies for ' + bot + ':'
print '-----------------------------------------------------------------'
for movie in topmov[bot]:
    print movie[1]
print "-----------------------------------------------------------"
print 'Least Recommended movies for' + bot + ':'
print '-----------------------------------------------------------------'
for movie in botmov[bot]:
    print movie[1]
print "-----------------------------------------------------------"
```

Results:

```
300 / 1664
400 / 1664
500 / 1664
600 / 1664
700 / 1664
800 / 1664
900 / 1664
1000 / 1664
1100 / 1664
1200 / 1664
1300 / 1664
1400 / 1664
1500 / 1664
1600 / 1664
100 / 1664
200 / 1664
300 / 1664
400 / 1664
500 / 1664
600 / 1664
700 / 1664
800 / 1664
900 / 1664
1000 / 1664
1100 / 1664
1200 / 1664
1300 / 1664
1400 / 1664
1500 / 1664
1600 / 1664
Best Recommended movies for Toy Story (1995):
------------------------------------------------------------
Wings of Courage (1995)
Wife, The (1995)
Visitors, The (Visiteurs, Les) (1993)
Van, The (1996)
Three Lives and Only One Death (1996)
------------------------------------------------------------
Least Recommended movies for Toy Story (1995):
------------------------------------------------------------
Aiqing wansui (1994)
All Things Fair (1996)
B. Monkey (1998)
Babyfever (1994)
Baton Rouge (1988)
------------------------------------------------------------
Best Recommended movies for Braveheart (1995):
------------------------------------------------------------
Wings of Courage (1995)
Wife, The (1995)
The Deadly Cure (1996)
Sexual Life of the Belgians, The (1994)
Of Love and Shadows (1994)
------------------------------------------------------------
Least Recommended movies forBraveheart (1995):
------------------------------------------------------------
Aiqing wansui (1994)
American Strays (1996)
August (1996)
B. Monkey (1998)
Ballad of Narayama, The (Narayama Bushiko) (1958)
------------------------------------------------------------
```

Date

Based on the results, none of them peak my interest. I googled searched Wings of courage, and how out it wasn't much related to Toy Story. Therefore, the recommendations are off completely. So, No, I don't like the resulting films recommendations.

# Reference:

https://github.com/arthur-e/Programming-Collective
Intelligence/blob/master/chapter2/recommendations.py