ASSIGNMENT 6

CS 432 Web Science

Mackenzie Kerchner

Mackenzie Kerchner

Contents

Mackenzie Kerchner

Problem 1

```
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").
```

Solution

```
myAge="25"
myGender="M"
myOccupation="student"
prefs = rec.loadMovieLens()
userMatches = []
substitute = "584"
```

Initialize some values and load up the prefs object with all the review data from u.data using the loadMovieLens function provided by the module we were instructed to use by the professor[1].

```
with open("movielens/u.user", "r") as userlist: #find matching users
    for line in userlist:
        #print line
        user, age, gender, occupation, zip = line.split("|")
        if (age == myAge) and (gender == myGender) and (occupation == myOccupation):
            userMatches.append(user)
            userMatches=userMatches[-3:]# grab the last 3 user matches made
print "matching users list:", userMatches# part 1
```

Open up u.user, parse out each attribute and check for similarity to my own, save those that are the same.

```
for user in userMatches:
    list = prefs[user]
    topList = sorted(list, key=lambda rating: rating[1])[-3:] #sorts by user rating
    botList = sorted(list, key=lambda rating: rating[1], reverse=True)[-3:] #reversed order
    print "user", user, "------------------"
    print "most liked movies"
    print topList
    print "least liked movies"
    print botList
```

Sorts user's preferences by review ratings and prints to console

Mackenzie Kerchner


```
matching users list: ['584', '727', '893']
user 584 ------------------
most liked movies
['Jurassic Park (1993)', 'Austin Powers: International Man of Mystery (1997)', 'Mystery Science Theater 3000: The Movie (1996)']
least liked movies
['Jean de Florette (1986)', 'Wallace & Gromit: The Best of Aardman Animation (1996)', 'E.T. the Extra-Terrestrial (1982)']
user 727 ------------------
most liked movies
['Cyrano de Bergerac (1990)', 'Mystery Science Theater 3000: The Movie (1996)', 'Bye Bye, Love (1995)']
least liked movies
['2001: A Space Odyssey (1968)', 'E.T. the Extra-Terrestrial (1982)', 'M*A*S*H (1970)']
user 893 ------------------
most liked movies
['Rumble in the Bronx (1995)', 'Event Horizon (1997)', 'Twister (1996)']
least liked movies
["Dante's Peak (1997)", 'Face/Off (1997)', 'Fair Game (1995)']
```

I chose user 584 because I like their top 3 liked movies, don’t think I’d like ‘jean de Florette, and have no strong feelings towards E.T. or any of the Wallace & Gromit movies.

Mackenzie Kerchner

Problem 2

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

Solution

```
correlated_list = []
with open("movielens/u.user", "r") as userlist: #find matching users
    for line in userlist:
        id, age, gender, occupation, zip = line.split("|")
        if id == substitute:
            continue
        current_cor = rec.sim_pearson(prefs, substitute, id)
        correlated_list.append([current_cor, id])

sorted_by_correlation = sorted(correlated_list)#[-5:] # sorted list of users correlated to substitute
least_correlated_users = sorted_by_correlation[::-1][-5:]# top 5 least correlated users: [::-1] reverses the list
most_correlated_users = sorted_by_correlation[-5:]# top 5 most correlated users

print "most correlated users and their correlation value", most_correlated_users# part 2
print "least correlated users and their correlation value", least_correlated_users# part 2
```

I used the existing simPearson function from the recommendations module to make a list of correlations between my substitute user and all other users. I then sorted that list and

```
most correlated users and their correlation value [[1.0, '915'], [1.0, '920'], [1.0, '929'], [1.0, '935'], [1.000000000000001, '219']]
least correlated users and their correlation value [[-1.0, '136'], [-1.0, '133'], [-1.0, '123'], [-1.0, '107'], [-1.0000000000000007, '765']]
```

Mackenzie Kerchner

Problem 3

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).

Solution

```
def topMatches(
    prefs,
    person,
    n=5,
    similarity=sim_pearson,
):
    '''
    Returns the best matches for person from the prefs dictionary.
    Number of results and similarity function are optional params.
    '''

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

I created a slightly different function called botMatches by removing the scores.reverse from topMatches and used both of them to create a list of top a bottom recommended movies

```
top_matches = []
bot_matches = []
top_matches = rec.topMatches(prefs, substitute, 5)
bot_matches = rec.botMatches(prefs, substitute, 5)
print top_matches
print bot_matches
```

The movies are listed by their id's and similarity values.

```
[(1.0000000000000001, '219'), (1.0, '935'), (1.0, '929'), (1.0, '920'), (1.0, '915')]
[(-1.0000000000000007, '765'), (-1.0, '107'), (-1.0, '123'), (-1.0, '133'), (-1.0, '136')]
```

References

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