2017

Assignment 7

CS 432

MIRANDA SMITH

Question 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? Choose the user that is most like you.

I chose 3 users that were similar to me by searching through u.users with awk for females around my age that were either students or programmers. I chose the following 3 users:

304|22|F|student|71701

599|22|F|student|R3T5K

300|26|F|programmer|55106 (youngest female programmer)

I wrote a simple python program 'get_sub_me.py' to find the 3 favorite and 3 least favorite for each user. I loaded the movies in with 'recomendations.py' which the code was retrieved from Programming Collective Intelligence written by Toby Segaran. I then retrieved the rated movies for each user and sorted them. I used a slice to get the top and bottom 3 movies for each user and printed them out.

```
import recomendations as r
import operator

pref= r.loadMovieLens()



AsortPref = sorted(pref['304'].items(),key=operator.itemgetter(1), reverse=True)
BsortPref = sorted(pref['599'].items(),key=operator.itemgetter(1), reverse=True)
CsortPref = sorted(pref['300'].items(),key=operator.itemgetter(1), reverse=True)

Atop3 = dict(AsortPref[:3])
Abottom3 = dict(AsortPref[-3:])
Btop3 = dict(BsortPref[:3])
Bbottom3 = dict(BsortPref[-3:])
Ctop3 = dict(CsortPref[:3])
Cbottom3 = dict(CsortPref[-3:])

print("\tUser 304:\n Favorite")
print(Atop3)
print("Least Fav")
print(Abottom3)

print("\tUser 599:\n Favorite")
print(Btop3)
print("Least Fav")
print(Bbottom3)

print("\tUser 300:\n Favorite")
print(Ctop3)
print("Least Fav")
print(Cbottom3)
```

The output changes for each run because more than 3 movies can be rated the same and it's unpredictable which 3 movies will be chosen. This is the output I used to choose the most similar user:

```
        User 304:
 Favorite
{'Face/Off (1997)': 5.0, 'Air Force One (1997)': 5.0, 'Jerry Maguire (1996)': 5.0}
Least Fav
{'Wishmaster (1997)': 2.0, 'English Patient, The (1996)': 1.0, 'George of the Jungle (1997)': 1.0}
        User 599:
 Favorite
{'Time to Kill, A (1996)': 5.0, "My Best Friend's Wedding (1997)": 5.0, 'Selena (1997)': 5.0}
Least Fav
{'For the Moment (1994)': 2.0, 'Event Horizon (1997)': 1.0, 'Love Jones (1997)': 2.0}
        User 300:
 Favorite
{'Money Talks (1997)': 5.0, 'Thin Line Between Love and Hate, A (1996)': 5.0, 'Love Jones (1997)': 5.0}
Least Fav
{'Fargo (1996)': 3.0, 'Mimic (1997)': 1.0, "McHale's Navy (1997)": 2.0}
```

My subsequent selection was user 304 because it was the only one that wasn't all romance movies.

Question 2

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

Using the topMatches function provided from recommendations.py I found the top 5 correlated users. I created another function in that program called bottomMatches which is the same as topMatches but doesn't reverse the list after sorting it to find the most negatively correlated users.

```
import recomendations as r

pref= r.loadMovieLens()
sub_me = '304'

similar_users = r.topMatches(pref,sub_me)
diff_users =  r.bottomMatches(pref,sub_me)
print("Top Matches:")
for item in similar_users:
    print(item[1])
print("Bottom Matches:")
for item in diff_users:
    print(item[1])
```

The results are as follows for user 304.

```
Top Matches:
31
914
844
794
785
Bottom Matches:
88
103
120
147
150
```

Question 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 are almost certain to hate).

From the functions provided I had 2 options for getting recommendations. getRecommendations calculates recommendations based on collaborative filtering. getRecommendedItems calculates recommendations based on content-based filtering. I chose to use getRecommendations because it was the one covered in class and thus the obvious choice for what the assignment would want.

I build a small python program to get the list of ranked movies for the user 304 and printed out the top and bottom 5.

```
import recomendations as r

pref= r.loadMovieLens()
sub_me = '304'

rankings = r.getRecommendations(pref, sub_me)

print("RECOMMENDATIONS")
for movie in rankings[:5]:
    print(movie[1])
print("HATED MOVIES")
for movie in rankings[-5:]:
    print(movie[1])
```

The output of that program is as follows.

```
RECOMMENDATIONS
Saint of Fort Washington, The (1993)
Two or Three Things I Know About Her (1966)
Star Kid (1997)
Someone Else's America (1995)
Santa with Muscles (1996)
HATED MOVIES
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)
```

Question 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?

I created another small python program to interact with the given code and created 'get_my_recommended_movies.py'. I chose Pulp Fiction as my favorite movie and My best friend's wedding as my least favorite movie. I used calculateSimilarItems to get a list of movies that were most like my picks. I also created calculateDiffItems as a copy of calculateSimilarItems but it calls bottomMatches instead of topMatches to get the least correlated movies.

```
import recomendations as r

pref= r.loadMovieLens()

fav_movie = "Pulp Fiction (1994)"
wor_movie = "My Best Friend's Wedding (1997)"

similar_movies = r.calculateSimilarItems(pref, 5)
diff_movies = r.calculateDiffItems(pref,5)

print(fav_movie)
print("\nMost Correlated")
for movie in similar_movies[fav_movie]:
    print(movie[1])
print("\nLeast Correlated")
for movie in diff_movies[fav_movie]:
    print(movie[1])
print("\n" + wor_movie)
print("\nMost Correlated")
for movie in similar_movies[wor_movie]:
    print(movie[1])
print("\nLeast Correlated")
for movie in diff_movies[wor_movie]:
    print(movie[1])
```

The results are as follows, excluding the output showing program progress from calculateSimilarItems and calculateDiffItems.

```
Pulp Fiction (1994)

Most Correlated
Á köldum klaka (Cold Fever) (1994)
Window to Paris (1994)
Wife, The (1995)
Wedding Gift, The (1994)
Two Friends (1986)

Least Correlated
American Strays (1996)
August (1996)
B. Monkey (1998)
Big Bang Theory, The (1994)
Bird of Prey (1996)

My Best Friend's Wedding (1997)

Most Correlated
Á köldum klaka (Cold Fever) (1994)
Wings of Courage (1995)
Widows' Peak (1994)
When Night Is Falling (1995)
Wedding Gift, The (1994)

Least Correlated
1-900 (1994)
8 Seconds (1994)
Aiqing wansui (1994)
Albino Alligator (1996)
All Things Fair (1996)
```

I agree with the results for Pulp Fiction, however the most correlated for My Best Friend's Wedding doesn't seem to match up. Cold Fever is not a romance at all and Wings of Courage is more of an adventure movie per the preview. I cannot say whether I dislike or like the movie because I haven't seen any of them, but from the previews I can say that I don't think I would like many of the movies correlated to Pulp Fiction, but they do seem to be similar to the movie itself.