Methods of data mining

Assignment 4

Task 3

Miro-Markus Nikula

712686

a) The mean values are calculated between all ratings of a user. The zeros (not seen movies) are not counted. The values are presented in the table below:

user	mean
U1	2
U2	3
U3	3
U4	4
U5	3.6
U6	3.3

b) Because the correlation coefficients are calculated by using only the movies that both users have rated, we can't use Numpy's corrcoef-method and because the means that are used are calculated from all the movies a user has rated, we can't use Scipy's pearsonr either. Therefore, I wrote a custom method for computing the Pearson's correlation coefficient that uses the means calculated from all movies. The Python code can be found in Task3.zip. To determine the similarities, I got the following results for the pairwise correlations:

	u1	u2	u3	u4	u5	u6
u1	1	0.816497	0.707107	1	-0.81111	-0.72058
u2	0.816497	1	0	1	-0.55902	-0.72058
u3	0.707107	0	1	0.316228	-0.58926	-0.55701
u4	1	1	0.316228	1	-0.68359	-0.37139
u5	-0.81111	-0.55902	-0.58926	-0.68359	1	0.904526
u6	-0.72058	-0.72058	-0.55701	-0.37139	0.904526	1

c) Below are the final user ratings for the movies. The previously missing and now predicted ratings are on **bold** text. For user 1 we were able to predict movie 5 rating by using the ratings of u1's two nearest neighbors u2 and u3:

(0.82*(4-3) + 0.71*(2-3)) / (0.82+0.71) + 2 ≈ **2.07.** This movie would be recommended to user 1 since its recommended rating is higher than the user's average rating.

For user 4 movies 1 and 6 were predicted using the ratings of u1 and u2. Because the correlation coefficients between u1 and u4 and u2 and u4 have a value of 1, the predictions are supposed to be as accurate as they get. The predictions were calculated as before, and I got predicted ratings for movie 1 = 5 and movie 6 = 3.5. The average rating of user 4 is 4, which means that movie 6 wouldn't be recommended.

Users 5 and 6 also have some missing ratings, but we can't predict those, since they both only have one correlation whose value is higher than the threshold 0.5. In fact, the rest of the correlation coefficients are negative.

Movie ratings by user:

	m1	m2	m3	m4	m5	m6
u1	3	1	2	2	**2.07**	2
u2	4	2	3	3	4	2
u3	4	1	3	3	2	5
u4	**5.0**	3	4	4	5	**3.5**
u5	2	5	5	0	3	3
u6	1	4	0	5	0	0

d) Predicting movie ratings using item-based way of adjusting cosine similarity wouldn't be such a good idea since cosine similarity doesn't work with zero vectors and with m3 and m4 we would encounter problems since there are four average ratings and then some zeros. Also, cosine similarity gives better results with positive values, but we would end up with many negative values especially with "bad" movies.

A more suitable way would most likely be to use graph-based methods to do the recommendations. This way, we would not use the ratings from 1 to 5, but we would only determine if the user likes the movie or not (0 or 1) by comparing to the average rating value. We can then calculate for example page rank for each movie and then determine the two most similar movies. This is a way to find whether the movie would be recommended to the user or not, but not the exact ratings of the movies.