3 Task3

The table below show the movie ratings by different users. Special value 0 means a missing rating.

	$m1$	$m2$	$m3$	$m4$	$m5$	$m6$
u1	3	1	2	2	0	2
u2	4	2	3	3	4	2
u3	4	1	3	3	2	5
u4	0	3	4	4	5	0
u5	2	5	5	0	3	3
u6	1	4	0	5	0	0

Table 1: Movie ratings (scale 1–5) by 6 users ($u1$–$u6$) on 6 movies ($m1$–$m6$).

3.1 Task3.a

After computation, the mean rating of each user are [**2 , 3 , 3 , 4 , 3.6 , 3.33**].

3.2 Task3.b

According to Pearson correlation in Aggarwal Equation 18.2, we can compute pairwise similarities between users shown in following table.

	$u1$	$u2$	$u3$	$u4$	$u5$	$u6$
u1	1.00	0.85	0.71	1.00	-0.82	-0.72
u2	0.85	1.00	0.00	1.00	-0.56	-0.72
u3	0.72	0.00	1.00	0.43	-0.59	-0.58
u4	1.00	1.00	0.43	1.00	-0.87	1.00
u5	-0.82	-0.56	-0.59	-0.87	1.00	1.00
u6	-0.72	-0.72	-0.58	1.00	1.00	1.00

Table 2: Pearson correlations between 6 users

3.3 Task3.c

```python
def predict(user,movie,NN,avg,sim):
    movies=np.array(records.iloc[:,movie])

    m1,m2=movies[NN[0]]-avg[NN[0]],movies[NN[1]] -avg[NN[1]]
    s1,s2=sim.iloc[user,NN[0]],sim.iloc[user,NN[1]]

    rating=(s1*m1+s2*m2)/(s1+s2)+avg[user]
    return round(rating,2), rating>avg[user]
```

Based on the average ratings and similarities obtained above, now we can predict the missing ratings using two nearest neighbours ($K = 2$) with similarity r > 0.5. The predicted rating are in following table. T means the predicted rating is bigger than user's average rating, we will recommend this movie to current user.

	m1	*m2*	*m3*	*m4*	*m5*	*m6*
u1	3	1	2	2	**3.0(T)**	2
u2	4	2	3	3	4	2
u3	4	1	3	3	2	5
u4	**5.0(T)**	3	4	4	5	**3.5(F)**
u5	2	5	5	**0**	3	3
u6	1	4	**4.03(T)**	5	**3.53(T)**	**0**

Table 3: Updated Movie ratings

We need to know not all missing ratings can be predicted in current conditions. we can't predict the ratings on m4 by u5, m6 by u6. Because when we are finding their nearest neighbours, we fail to find enough 2 NNs with similarity $r > 0.5$, and the NNs already rated the movie we are predicting.

3.4 Task3.d

Now, we consider the item-based way of predicting the missing ratings of movies with adjusted cosine similarity. The formula shows below,

$$Cosine(\bar{U}, \bar{V}) = \frac{\sum_{i=1}^{s} u_i \cdot v_i}{\sqrt{\sum_{i=1}^{s} u_i^2} \cdot \sqrt{\sum_{i=1}^{s} v_i^2}} \tag{9}$$

$U = (u_1, ..., u_s)$ and $V = (v_1, ..., v_s)$ of a pair of items' normalized ratings.

The adjusted cosine similarity of m3 and m4 are 1,because the movies ratings are same if not consider which users. So it means, with cosine similarity the model will give very similar predictions for different movies for different users, because they have too many similar watched records. This makes the differences between different users not reflected, making the whole system more and more homogeneous.

An alternative item-based solution is increasing some randomness when facing too much similarity. When the recommended movies become more and more similar, we can recommend some random movies in addition to the candidate movies. Through users' feedback on random movies, i.e. whether they choose to watch or not or how much they rate, we add diversity to our recommendation system and the recommended results are more personalized.