Assignment 3 Task2
Qi Sun

- **Explain how the solution in the provided report predicts the missing values in theCollaborative Filtering by using your own language clearly and completely.**

First of all, for the original rating matrix R, if a score $r_{u,i}$ is missing, then starting with the user and the item. And looking for users who are similar to this user and items that are similar to this item, and then find similar user names and similar item ratings to help predict this missing value. Using similar users, we can get a prediction score called score_u, and using similar items we can get a prediction score called score_i. Therefore, the predicted value will be score_u when user has no similar users and the predicted value will be score_i when this item has no similar items. So, the whole thing can be divided into three steps.

Step1: Calculate similar items and similar users

Since we want to find similar users and similar items, it is necessary to calculate the similarity between users and items. The similarity between them will be judged as similarity when they reach a certain threshold.

User similarity is calculated as follows:

$$\mathrm{Sim}(a,u)=\frac{\sum_{i\in I(a)\cap I(u)}\left(r_{a,i}-\bar{r}_a\right)\cdot\left(r_{u,i}-\bar{r}_u\right)}{\sqrt{\sum_{i\in I(a)\cap I(u)}\left(r_{a,i}-\bar{r}_a\right)^2}\cdot\sqrt{\sum_{i\in I(a)\cap I(u)}\left(r_{u,i}-\bar{r}_u\right)^2}}$$

$$\mathrm{Sim}'(a,u)=\frac{\mathrm{Min}(|I_a\cap I_u|,\gamma)}{\gamma}\cdot\mathrm{Sim}(a,u)$$

$Sim'(a,u)$ is the final similarity.

The similarity of the item is calculated as follows:

$$\mathrm{Sim}(i,j)=\frac{\sum_{u\in U(i)\cap U(j)}\left(r_{u,i}-\bar{r}_i\right)\cdot\left(r_{u,j}-\bar{r}_j\right)}{\sqrt{\sum_{u\in U(i)\cap U(j)}\left(r_{u,i}-\bar{r}_i\right)^2}\cdot\sqrt{\sum_{u\in U(i)\cap U(j)}\left(r_{u,j}-\bar{r}_j\right)^2}}$$

$$\mathrm{Sim}'(i,j)=\frac{\mathrm{Min}(|U_i\cap U_j|,\delta)}{\delta}\cdot\mathrm{Sim}(i,j)$$

$Sim'(i,j)$ is the final similarity.

Then evaluate the user and the item based on the threshold

$$S(u) = \left\{ u_a \mid \mathrm{Sim}'(u_a, u) > \eta, u_a \neq u \right\}$$

$$S(i) = \left\{ i_k \mid \mathrm{Sim}'(i_k, i) > \theta, i_k \neq i \right\}$$

Step 2: The prediction scores score_u and score_i were calculated according to similarusers and similar items

$$score\_u = \bar{u} + \frac{\sum_{u_a \in S(u)} \mathrm{Sim}'(u_a, u) \cdot (r_{u_a,i} - \bar{u}_a)}{\sum_{u_a \in S(u)} \mathrm{Sim}'(u_a, u)}$$

$$score\_i = \bar{i} + \frac{\sum_{i_k \in S(i)} \mathrm{Sim}'(i_k, i) \cdot (r_{u,i_k} - \bar{i}_k)}{\sum_{i_k \in S(i)} \mathrm{Sim}'(i_k, i)}$$

Step 3: Select the predicted value based on whether there are similar users or items

when $S(u) \neq \emptyset and S(i) = \emptyset$:

predict=score_u

when $S(i) \neq \emptyset and S(u) = \emptyset$:

predict=score_i

when $S(i) \neq \emptyset and S(u) \neq \emptyset$:

$$predict = \lambda \times score\_u + (1 - \lambda) \times score\_i$$

- **Explain why the solution in the provided report can tackle the missing value problem in Collaborative Filtering clearly and completely**

  The main point is to solve the problem of data sparse. In the report, the previous work has clustering based smoothing algorithm, but the clustering based smoothing algorithm limits the diversity of users in each cluster, and it predicts that all missing data in the user-item matrix will bring negative impact on active user recommendation. The importance is the weighting method based on Pearson Correlation Coefficient. It is used to calculate the similarity between users and items, which overcomes the problem that the similarity accuracy may decline. In addition, missing values are predicted based on this method, and only when the recommendation of active users has a positive impact, instead of predicting every missing piece of data in the User-Item matrix, the algorithm in the report predicts missing data in the user-Item matrix. Therefore, this method has good performance to tackle the missing value.

- ## Explain how you implement the solution clearly and completely

  My implementation, as I described in the first question, is divided into three steps.

  Step 1: Calculate the similarity. I use a function to do this

```
def simUser(data,para,threshold):
  '''
  '''
  '''
return User_User,user_mean
```

  The parameters passed in are the original rating matrix and the superparameter. The similarity matrix of users and the average rating list of each user are returned

```
def simItem(data,para,threshold):
  '''
  '''
  '''
  return item_item,item_mean
```

  The parameters passed in are the original scoring matrix and the superparameter. The similarity matrix of items and the average scoring list for each item are returned

Step 2: Calculate user - and item-based scores, respectively

Its calculation is described by the formula, and the key codes are as follows:

```
                    A=0
                    B=0
                    for each in sim_user_u:
                        if each!=u:
                            A+=user2user[u][each]*(rating_matrix[each][i]-
user_mean[each])
                            B+=user2user[u][each]
                    u_score+=user_mean[u]+A*1.0/B
```

```
                    A=0
                    B=0
                    for each in sim_item_i:
                        if each!=i:
                            A+=item2item[i][each]*(rating_matrix[u][each]-
item_mean[each])
                            B+=item2item[i][each]
                    i_score+=item_mean[i]+A*1.0/B
```

Step 3: Calculate the final predicted value according to the situation.