Report for hw4

2018000337 장호우

Environment

- Windows 10
- Python 3.7.7 64-bit

Requirements

- numpy
- pandas

To install requirements

```
pip3 install numpy
pip3 install pandas
```

Content

Summary of algorithm

User-Based Collaborative Filtering which is a memory-based method. That people with similar characteristics share similar taste. The method identifies users that are similar to the queried user and estimate the desired rating to be the weighted average of the ratings of these similar users. Thus, predict users' ratings.

Using the ratings of other people similar to the user. The formula I have used is,

$$s(u,i) = \bar{r}_u + \frac{\sum_{v \in V}(r_{vi} - \bar{r}_v) * w_{uv}}{\sum_{v \in V} w_{uv}}$$

Where `s` is the predicted score, `u` is the user, `i` is the item, `r` is the rating given by the user and `w` is the weight. I used the cosine similarity to calculate the weight given in the above formula.

Detailed description of each function

```
class UserCF:
    def __init__(self, train_data):
        self.train = train_data
        self.n_users = max(self.train.iloc[:, 0].unique())
        self.n_items = max(self.train.iloc[:, 1].unique())
        self.rating_mat = np.full((self.n_users, self.n_items), -1)
        self.avg_rating = np.zeros(self.n_users)
        self.user_sim_mat = np.ones([self.n_users, self.n_users])

        # to build User-Item matrix
        for user_id, item_id, rating, _ in self.train.values.tolist():
            self.rating_mat[user_id-1][item_id-1] = rating
```

```
        # to calculate the avg rating of each user
        for user in range(self.n_users):
            rated_item = (self.rating_mat[user] >= 0)
            self.avg_rating[user] = self.rating_mat[user][rated_item].sum() /
len(rated_item[rated_item])

        # to build USER-USER similarity matrix
        for i in range(self.n_users):
            for j in range(self.n_users):
                self.user_sim_mat[i][j] = self.cosine(self.rating_mat[i],
self.rating_mat[j])
```

```
# to calculate cosine similarity
@staticmethod
def cosine(vec1, vec2):
    mat_a, mat_b = np.mat(vec1), np.mat(vec2)
    denom = np.linalg.norm(mat_a) * np.linalg.norm(mat_b)
    return (0.5 + 0.5 * (float(mat_a * mat_b.T) / denom))
```

```
def predict(self, test_data):
    res = list()
    for user_id, item_id, _, _ in test_data.values.tolist():
        user, item = user_id-1, item_id-1
        try:
            item_info = self.rating_mat[:, item]
            rated = (item_info >= 0)
            rated_user_sim = self.user_sim_mat[user, rated]
            # if the user have no similar user
            if rated_user_sim.sum() == 0:
                res.append([user_id, item_id, self.avg_rating[user]])
            else:
                # using USER-USER similarity matrix to predict users' ratings
                user_sim_ratings = rated_user_sim * (item_info[rated] -
self.avg_rating[rated])
                rating = self.avg_rating[user] + user_sim_ratings.sum() /
rated_user_sim.sum()
                res.append([user_id, item_id, np.clip(rating, 1, 5)])
        except IndexError as e:
            # if new user
            res.append([user_id, item_id, self.avg_rating[user]])
    return res
```

How to run it

You could click the `run.bat` file to run and test all five examples on Windows 10. The testing result will be saved to `res.txt` file.

The testing results as follow:

RMSE

Example	u1	u2	u3	u4	u5
RMSE	0.9732217	0.9636413	0.9564762	0.9538846	0.9541221

Running Each Example

For Example 1~5

```
python recommender.py u1.base u1.test
python recommender.py u2.base u2.test
python recommender.py u3.base u3.test
python recommender.py u4.base u4.test
python recommender.py u5.base u5.test
```

Testing Each Example

For Example 1~5

```
./PA4.exe u1
./PA4.exe u2
./PA4.exe u3
./PA4.exe u4
./PA4.exe u5
```

File Description

Train / Test files Testing program	Prediction result files	Documents
PA4.exe run.bat u1.base u2.base u3.base u4.base u5.base u1.test u2.test u3.test u4.test u5.test	u1.base_prediction.txt u2.base_prediction.txt u3.base_prediction.txt u4.base_prediction.txt u5.base_prediction.txt	2020_DM_Programming_Assignment_4.pdf README.md report4.pdf res.txt