Recommender System

author DaeIn Lee

Quick Start

All programs are tested on macOS 10.14

Requirements

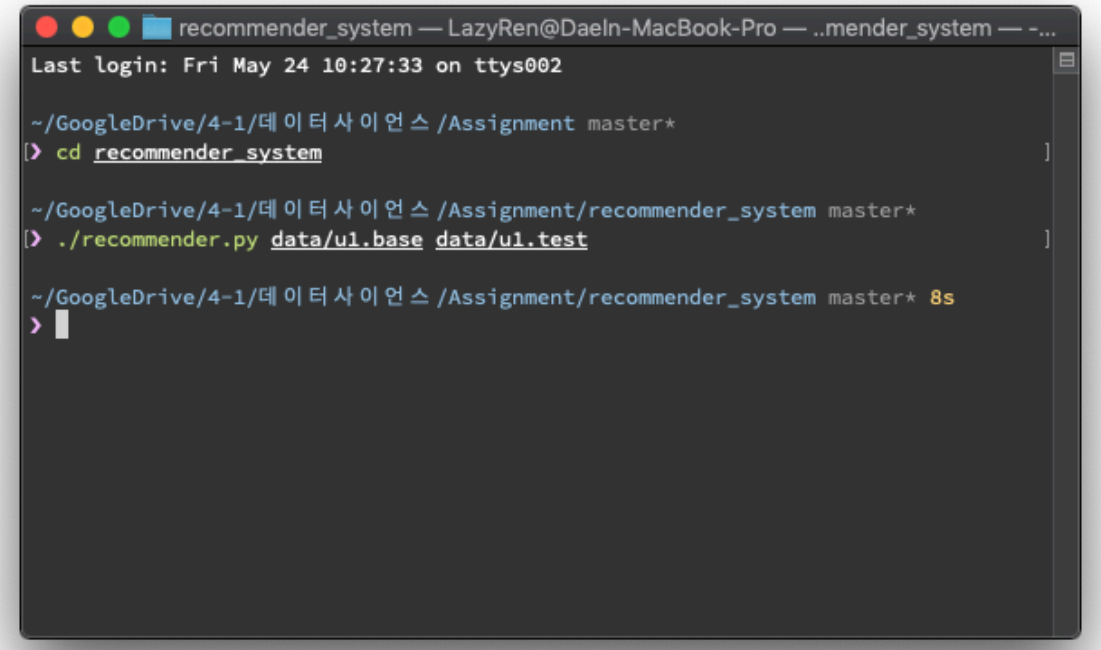
- Python3

How to Run

recommender.py

```
./recommender.py [train file] [test file]
python3 recommender.py [train file] [test file]
```

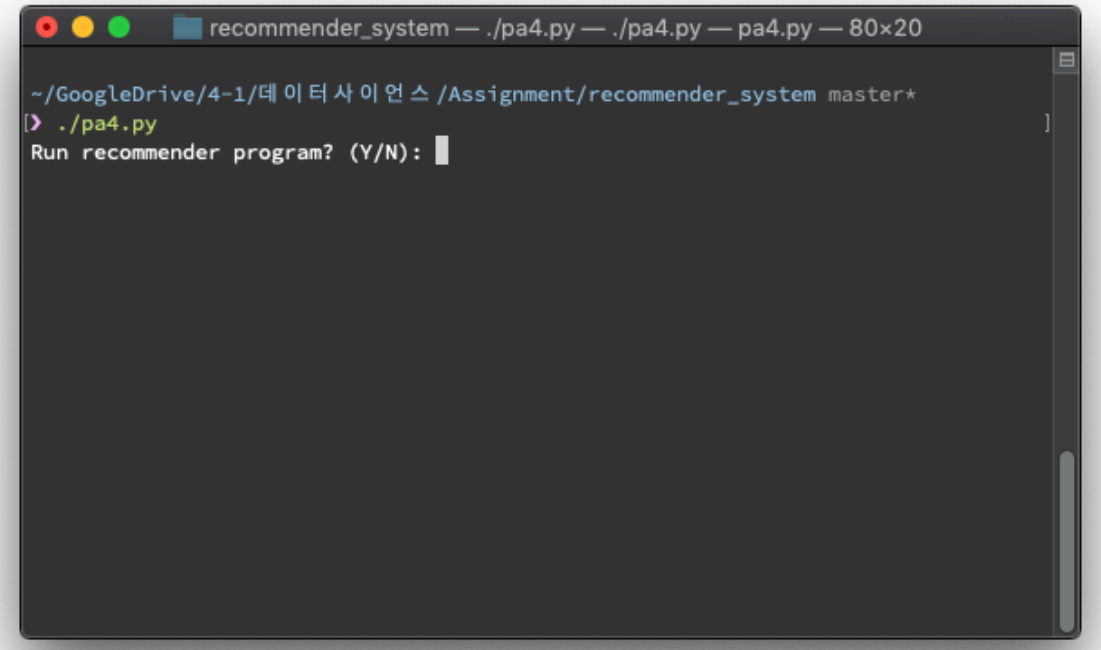
In order to execuate program using first command, you must have execution permission for `recommender.py`.
If it gives permission error, either give it an execution permission or use second command.



pa4.py

```
./pa4.py
python3 pa4.py
```

Running above command will ask you whether to run recommender program or not.



If you choose to run recommender program, `pa4.py` will automatically run `recommender.py` to train & test all u1 ~ u5.

```
recommender_system — LazyRen@DaeIn-MacBook-Pro — ..mender_system — -zsh ▸ -zsh — 95×39
~/GoogleDrive/4-1/데이터사이언스/Assignment/recommender_system master*
❯ ./pa4.py
Run recommender program? (Y/N): y
Running u1 Test
Prediction took 8.579507 seconds
Starting Evaluation: u1
RMSE: 0.9957409 Total: 19830
[(-4, 5), (-3, 39), (-2, 768), (-1, 5293), (0, 8332), (1, 4156), (2, 1171), (3, 226), (4, 10)]
Evaluation took 0.010867 seconds

Running u2 Test
Prediction took 8.749938 seconds
Starting Evaluation: u2
RMSE: 0.9888124 Total: 19555
[(-4, 4), (-3, 73), (-2, 712), (-1, 5177), (0, 8468), (1, 4203), (2, 1149), (3, 202), (4, 12)]
Evaluation took 0.010367 seconds

Running u3 Test
Prediction took 8.808206 seconds
Starting Evaluation: u3
RMSE: 0.9865343 Total: 19465
[(-4, 3), (-3, 42), (-2, 794), (-1, 5264), (0, 8412), (1, 4178), (2, 1081), (3, 217), (4, 9)]
Evaluation took 0.010925 seconds

Running u4 Test
Prediction took 8.984601 seconds
Starting Evaluation: u4
RMSE: 0.9804591 Total: 19226
[(-4, 3), (-3, 52), (-2, 736), (-1, 5255), (0, 8453), (1, 4169), (2, 1139), (3, 186), (4, 7)]
Evaluation took 0.010812 seconds

Running u5 Test
Prediction took 8.984318 seconds
Starting Evaluation: u5
RMSE: 0.9813511 Total: 19261
[(-4, 1), (-3, 36), (-2, 750), (-1, 5254), (0, 8335), (1, 4282), (2, 1147), (3, 189), (4, 6)]
Evaluation took 0.011136 seconds
```

Else, `pa4.py` will use existing u`N`.test & u`N`.base_prediction.txt files to evaluate **RMSE**.

You **MUST** have `base_prediction.txt` files ready to run in this way.

```
recommender_system — LazyRen@DaeIn-MacBook-Pro — ..mender_system — -zsh ▸ -zsh — 95×28
~/GoogleDrive/4-1/데이터사이언스/Assignment/recommender_system master* 53s
❯ ./pa4.py
Run recommender program? (Y/N): n
Starting Evaluation: u1
RMSE: 0.9957409 Total: 19830
[(-4, 5), (-3, 39), (-2, 768), (-1, 5293), (0, 8332), (1, 4156), (2, 1171), (3, 226), (4, 10)]
Evaluation took 0.010272 seconds

Starting Evaluation: u2
RMSE: 0.9888124 Total: 19555
[(-4, 4), (-3, 73), (-2, 712), (-1, 5177), (0, 8468), (1, 4203), (2, 1149), (3, 202), (4, 12)]
Evaluation took 0.010822 seconds

Starting Evaluation: u3
RMSE: 0.9865343 Total: 19465
[(-4, 3), (-3, 42), (-2, 794), (-1, 5264), (0, 8412), (1, 4178), (2, 1081), (3, 217), (4, 9)]
Evaluation took 0.01091 seconds

Starting Evaluation: u4
RMSE: 0.9804591 Total: 19226
[(-4, 3), (-3, 52), (-2, 736), (-1, 5255), (0, 8453), (1, 4169), (2, 1139), (3, 186), (4, 7)]
Evaluation took 0.010186 seconds

Starting Evaluation: u5
RMSE: 0.9813511 Total: 19261
[(-4, 1), (-3, 36), (-2, 750), (-1, 5254), (0, 8335), (1, 4282), (2, 1147), (3, 189), (4, 6)]
Evaluation took 0.010379 seconds
```

Detail of the execution, such as directory of data files, recommender program's name or which tests to run can be modified by changing below globar variables.

```
inputFileDir   = "data/"
idealFileDir   = "data/"
outputFileDir  = "data/"
EXECUTABLE_NAME = "recommender.py"
testNameList    = ["u1", "u2", "u3", "u4", "u5"]
```

Implementation

loadData()

Simply read line to line from *fileName* and generate list of *[(user_id), (item_id), (rating)]*
Note that *time_stamp* has been removed when parsed. Since prediction does not use that information in any way with current implementation.

preprocessData()

Preprocess train data that has been loaded by `loadData()`.
For now, it only creates *ratingDict* for the future use.

```
ratingDict[user][movie]  # rating of moive of user (user & movie must be int)
ratingDict[user]['mean'] # mean of user's all rating
```

Time Complexity = O(2r + u)
r = number of rows in data, u = number of users

similarityMeasure()

$$\text{simil}(x, y) = \frac{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r_x})(r_{y,i} - \bar{r_y})}{\sqrt{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r_x})^2} \sqrt{\sum_{i \in I_{xy}} (r_{y,i} - \bar{r_y})^2}}$$

Calcurate similarity using *Pearson Correlation Coefficient*.
Function only needs `ratingDict` created by `preprocessData()`.

Time Complexity = O(u^2 * m)
u = number of users, m = number of movies
Since m in time complexity refers to commonItem of two user, in most cases function will run in O(u^2).

findNeighbors()

Calculate similarity of all user and sort them in descending order of similarity.
Return 2D list of tuple(uid, smilarity).

```
# Returned 2D list
neighbors[uid][i] == (uid2, sim) # uid's ith closest neighbor who's id is
uid2, with similarity of sim.
```

Time Complexity = O(u^2)
u = number of users

predictRating()

$$pred_{u,i} = \overline{r}_u + \frac{\sum_{v \in V} sim_{u,v}(r_{v,i} - \overline{r}_v)}{\sum_{v \in V} |sim_{u,v}|}$$

Predict rating of movie for user using KNN collaborative filtering.
Prediction use the mean of the neighbouring ratings weighted by their similarity[1].
Algorithm stops in advance if similarity reaches negative for better prediction.

Time Complexity = O(u)
u = number of users

---

1. Florent Garcin, Boi Faltings, Radu Jurca, Nadine Joswig, **Rating Aggregation in Collaborative Filtering Systems**↩