

CSE 4334/5334 – Data Mining

Fall 2014 - Project 1

Due: 11:59pm Central Time, Saturday, Nov 15, 2014

Submitted By: Sarvesh Sadhoo (1000980763)

Implementation Idea:

For the purpose of implementing the project the idea to build the classifier was obtained from the black board discussion given by the TA. The idea stated as below:

For each user u in U_2 , find its k -nearest-neighbor users. Assign a score to each job applied by u 's neighbor. Then pick top m jobs by the scores as the potential applications for user u . Then rank potential applications of all users to get top 150 results.

Design and Implementation:

Following steps were implemented for implementing the above-mentioned idea:

Step 1:

In this step the users.tsv file was parsed and a hash map was created with key as user_id and user attributes [state, country, degree, major, experience, current_employed] as a list of user attributes:

Hash Table Example: {47: [CA, US, MBA, Business, 23, 23]}

Step 2:

In this step the users2.tsv file was parsed and a list was created that contained all the user_id in users2.tsv

List Example: [1400836, 679942, 1032210, 180243]

Step 3:

In this step similarity was computed between users in user2.tsv and users.tsv. Similarity was calculated on the attributes [state, country, degree, major, experience, current_employed]. Score of 1 was assigned to every possible match for each attribute.

So max score of 6 and min score of 0 was possible as only 6 attributes were matched. The output of this step was a hash map of the following type:

```
{userid<104060>: {score<6>: [list of userid from users.tsv ],
                  score<5>: [list of userid from users.tsv ]}
}
```

Step 4:

In step 4 the above computed hash table was reduced. Only the top two score for each user id was taken and rest scores were dropped, as higher score were more similar to the users

Step 5:

In this step the jobs.tsv file was parsed and store in a hash table with key as job_id and value as job end date. In this hash table only those jobs where included which had an end date time greater than 2012-04-09 00:00:00.

Step 6:

In this step the apps.tsv file was parsed and a hash table was created with key as uid and value as list of job ids but with only those job ids that were computed in above step so that jobs with end date time lesser than 2012-04-09 00:00:00 were not included.

Hash Table Example: {userid: [list of job ids]}

Step 7:

In this step the hash table generated in step 3 was used to replace the userid list with a jobid list. The job id list was obtained from the above step.

Hash Table Example: {userid: score: [list of job ids]}

Step 8:

In this step the number of repetitive job ids were counted in the job list and as has was created as below:

```
{userid: {score: {job_id : frequency_score}}}
```

The hash table was a hash inside a hash inside hash.

Step 9:

In this step the above-generated hash table was used and a final score was computed by multiplying the frequency score of the job_id and the actual score for that particular user id.

Hash Table Example: {userid: {score1: [Job id list]
{score2: [job id list]}}

Step 10:

In this step the above hash only picking the top most score and its associated values reduced table.

Hash Table Example: {userid1: {score1: [Job id list]}}
userid2: {score1: [job id list]}}

Step 11:

In this step the above hash table generated in Step 10 is sorted according to the score to obtain the top 150 (UserID, JobID) pairs in the order of how likely a UserID will apply to a JobID.

Step 12:

The top 150 (UserID, JobID) pairs are written to the output.tsv file

Execution Steps:

1. Copy the python file (project2.py) included in the zip folder to the desired location on your machine.
2. Open your terminal/command prompt.
3. To run the project2.py run give the command in the below format.

Format:

```
python project2.py /p/t/users.tsv /p/t/user2.tsv /p/t/jobs.tsv /p/t/apps.tsv /p/t/output.tsv
```

Example:

```
python project2.py '/Users/srv/Desktop/Data Mining/Project 2/users.tsv'  
'/Users/srv/Desktop/Data Mining/Project 2/user2_1.tsv' '/Users/srv/Desktop/Data  
Mining/Project 2/jobs.tsv' '/Users/srv/Desktop/Data Mining/Project 2/apps.tsv' 'Output.tsv'
```

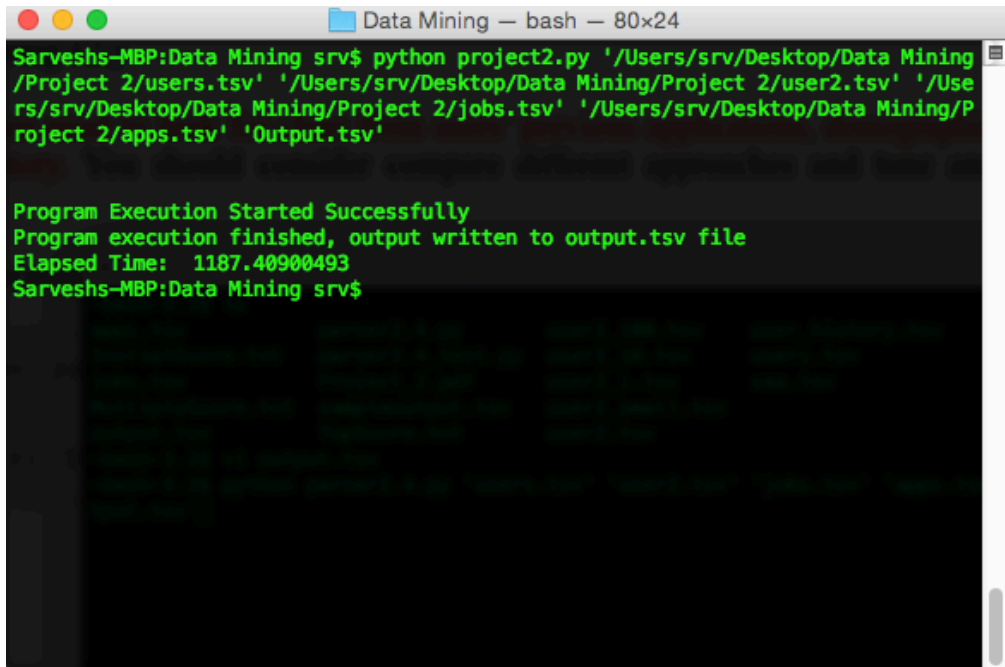
/p/t refers to the path of the input file

Note:

1. Make sure you give the path variables as a string. Also the input file sequence should also be same as shown in the example for the program to execute properly.
2. Make sure any empty output.tsv already exists.

Execution Screen Shot:

Program Execution On Local Machine:

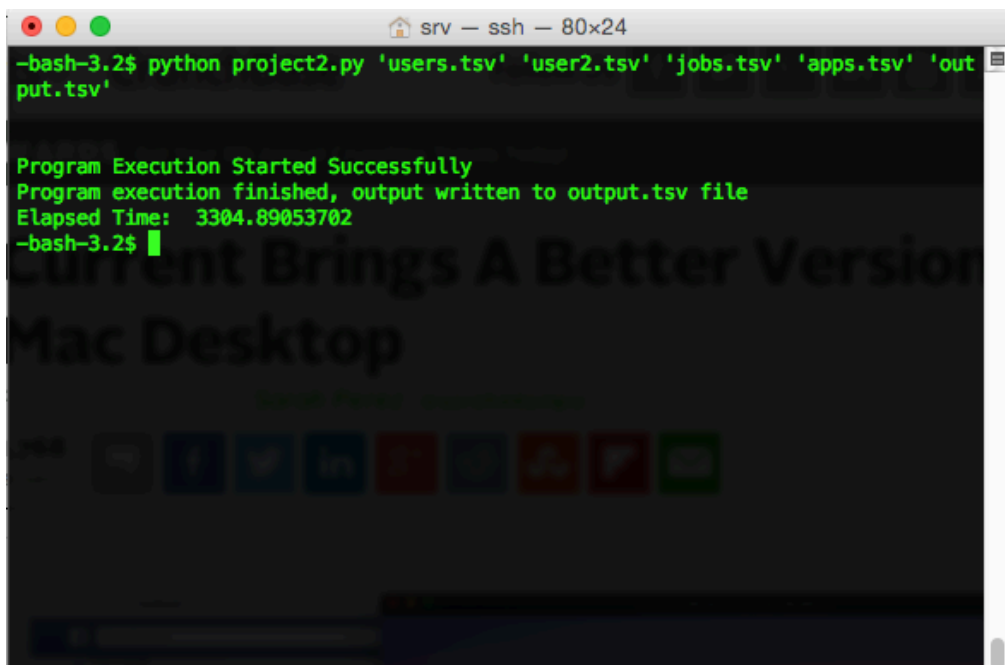


A terminal window titled "Data Mining — bash — 80x24" showing the execution of a Python script. The prompt is "Sarveshs-MBP:Data Mining srv\$". The command entered is "python project2.py '/Users/srv/Desktop/Data Mining/Project 2/users.tsv' '/Users/srv/Desktop/Data Mining/Project 2/user2.tsv' '/Users/srv/Desktop/Data Mining/Project 2/jobs.tsv' '/Users/srv/Desktop/Data Mining/Project 2/apps.tsv' 'Output.tsv'". The output shows "Program Execution Started Successfully", "Program execution finished, output written to output.tsv file", and "Elapsed Time: 1187.40900493". The prompt returns to "Sarveshs-MBP:Data Mining srv\$".

```
Sarveshs-MBP:Data Mining srv$ python project2.py '/Users/srv/Desktop/Data Mining/Project 2/users.tsv' '/Users/srv/Desktop/Data Mining/Project 2/user2.tsv' '/Users/srv/Desktop/Data Mining/Project 2/jobs.tsv' '/Users/srv/Desktop/Data Mining/Project 2/apps.tsv' 'Output.tsv'

Program Execution Started Successfully
Program execution finished, output written to output.tsv file
Elapsed Time: 1187.40900493
Sarveshs-MBP:Data Mining srv$
```

Program Execution On Omega Machine:



A terminal window titled "srv — ssh — 80x24" showing the execution of a Python script. The prompt is "-bash-3.2\$". The command entered is "python project2.py 'users.tsv' 'user2.tsv' 'jobs.tsv' 'apps.tsv' 'output.tsv'". The output shows "Program Execution Started Successfully", "Program execution finished, output written to output.tsv file", and "Elapsed Time: 3304.89053702". The prompt returns to "-bash-3.2\$".

```
-bash-3.2$ python project2.py 'users.tsv' 'user2.tsv' 'jobs.tsv' 'apps.tsv' 'output.tsv'

Program Execution Started Successfully
Program execution finished, output written to output.tsv file
Elapsed Time: 3304.89053702
-bash-3.2$
```