**310CT Intelligent Agents Assignment 1 (Individual Assignment)**

**Case: Recommendation system**

An online shoe store plans to develop a recommendation system. The system should:

1. actively acquire the user preference information,
2. acts as a server to provide the recommendation information per request.

You are required to build a prototype of the recommendation system using Java or Python programming language. File interface will be adopted in the prototype to simulate both actions mentioned above. So, the prototype will process different input files and produce the corresponding output files whenever appropriate. There are two types of input files, one is user preference files and the other is recommendation request files. The requirements are as follows:

1. **Recommendation Model Updating (user preference file)**

Simulation: The agent collects information from users actively and users give user preference files as the feedback.

Input file

* Directory: <application path>/input
* Filename: UPyyyymmddhhmmss.csv
* The file contains three columns in a row separated by commas, (userID, itemID, score [1.0-5.0])

Processed input file

* Directory: <application path>/processed
* Filename: UPyyyymmddhhmmss.csv
* The input file should be moved to the above directory.

Processing requirement

* After processing, the updated information should be reflected in the result of the recommendation request processing shortly, e.g. the data model to be used by the recommendation request processing should be updated accordingly.
* yyyymmddhhmmss is the timestamp of the request. The agent should handle the oldest user preference file first.
* Processed file should be moved to the directory <application path>/processed.

1. **Recommendation Request Processing (recommendation request file)**

Simulation: An agent receives a recommendation request file and it should try to provide a recommendation by creating a recommendation file as soon as possible.

Input file

* Directory: <application path>/input
* Filename: RQyyyymmddhhmmss.csv
* The file may contain multiple rows, each row contains one column (userID).

Output file

* Original file
  + Directory: <application path>/processed
  + Filename: RQyyyymmddhhmmss.csv
  + The input file should be moved to another directory.
* Response file
  + Directory: <application path>/output
  + Filename: RQyyyymmddhhmmssout.csv
  + The file contains two columns in a row (userID and a recommended item list separated by commas). The first column should be the same as the one in the input file.

Processing requirement

* Appropriate recommendation logic such as collaborative filtering logic should be used. Any appropriate library can be adopted.
* yyyymmddhhmmss is the timestamp of the request. The agent should handle the oldest request first.

1. **Other requirements**

* Two types of files should be handled separately whenever possible.
* For the processing of the recommendation request file, the response time and the processing time should be as short as possible.
* The information in the user preference file should be updated into the system as early as possible.
* If there is any conflict, the recommendation file should be processed first.
* The following processing information should be provided either in a log file or to the output console. Other useful information can be added, but no debug message should be provided with this information.

14:23:12.254 Model updating start: UPyyyymmddhhmmss.csv

14:23:14.250 Model updating end: UPyyyymmddhhmmss.csv

14:23:17.324 Request processing start: RQyyyymmddhhmmss.csv

14:23:17.854 Request processing end: RQyyyymmddhhmmss.csv

* Maintain a file to reflect the content of the most up-to-date model of the user preference information.

Filename: d:/310CT/processed/currentModel.csv

Content format: The same format as the user preference file, i.e. the file contains three columns in a row separated by commas, (userID, itemID, score [1 - 5]).

1. **Recommendation Evaluation (Bonus)**

In order to verify the recommendation module, a proper evaluation (e.g. root-mean-square error) can be conducted. Maximum 10% bonus will be rewarded if a proper evaluation has been conducted and discussed in the report.

**Marking Criteria**

The assessment will be according to the following criteria:

1. Proper implementation of progress control (35%)
   1. The efficiency, sequence and file movement have been handled properly.
   2. Proper priority control of file uploading and model updating.
2. Proper implementation of recommendation logic (35%)
   1. Proper recommendation logic implemented.
   2. Different scenarios can be supported with a reasonable response time.
3. Provision of design documentation to show (25%)
   1. How to start the program
   2. Overall program flow
   3. Recommendation feature discussion including library discussion (strength and weakness)
   4. Other libraries discussion
   5. Test result examples

(Maximum 4 pages for a-c with font size 12)

1. Explanation of the program
   1. Explanation of the program in the demonstration. (5%)
2. Bonus (Optional): Evaluation of the recommendation module (program + document with max. 2 pages)

**Submission Date**

13 Jul 2020 23:55

**Notes on submission**

* The following artifacts should be submitted on or before the submission date.
  + The design documentation and its Turnitin report.
  + For Java development (Eclipse):

Export the whole project in Eclipse into a zip file. (e.g. file -> export -> general -> archive file -> (select only your project) in Eclipse).

* + Any other programs
* This is an individual assignment, and submissions should be the work of one single student.
* You are required to demonstrate the system in the Lab on the submission date.

**Contribution**

Assignment 1: 50% (this assignment)

Assignment 2: 50%

Marking guideline:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | 40% | 70% |
| A | (25%) Process control - efficiency | * The flow of the updating logic and the recommendation logic are not complete, stable or smooth. (15%) * Fair efficiency program. (overall) (10%) | * High efficiency program, infinite loop, multiple demon, automation, proper process arrangement (15%) * Well-test efficiency program (overall) (10%) |
| B | (10%) Process control – basic file handling | * Some errors in the file handling sequence, file movement, file naming, no error handling (10%) | * Proper file handling sequence, file movement, file naming and proper error handling (10%) |
| C | (15%) Recommendation logic – model updating | * Fair recommendation model updating, data model with some errors (15%) | * Proper recommendation model updating, data model (15%) |
| D | (20%) Recommendation logic – usage | * Proper recommendation logic, library used (10%) * High efficiency (recommendation) (10%) | * Proper recommendation logic, library used (10%) * High efficiency (recommendation) (10%) |
| E | (25%) Documentation | * fair system design and program control flow (10%) * not much library explanation (10%) * fair program invoking instruction, and test result (5%) * Fair comments in the program (5%) | * Completeness system design and program control flow (10%) * Library explanation (10%) * Program invoking instruction, and test result (5%) * Clear comments in the program (5%) |
| F | (5%) Program explanation | * Not clear explanation of the program (5%) | * Clear explanation of the program (5%) |