Evaluating Dataset portions based on query logs

Data Mining, University of Trento, 2022-2023
Instructor: Prof. Yannis Velegrakis

Course project. DEADLINE: January 31st , 2023

The goal of this work is to design, develop, implement a sophisticated query recommendation system.

We assume that we have a database consisting of one single relational table (e.g., Person(id, name, address, age, occupation)). The table has a number of attributes and is populated with a great deal of tuples. A query is a conjunction of a set of conditions of the form Attribute=value. For example, a query can be name=John AND age=10. Another query can be age=25 AND address=Trento AND occupation=director).
We assume that we have a set of users and for each user we have a set of queries that the user has posed to the database. When a user poses a query to the database, he/she retrieves a set of tuples as an answer (that is the answer set). If he/she likes this set of tuples, he/she gives them a high score, otherwise he/she gives a low score.

The goal of this work is to create a query recommendation system which recommends to the user a set of queries, the answer set of which contains data that are of interest to the user. A query that is recommended for a user is a query that the specific user has not posed to the database in the past and is expected that if posed, the user will give a high score. Note that although the query may have not been posed by the specific user in the past, some of its tuples may have in the past been seen by the user, maybe because they were in the answer set of another query that the user has posed.

Part A:
The solution/tool that needs to be developed takes as input:

1. A relational table populated with tuples. The table is expressed as a CSV file, where each row is a tuple, and the first row contains the names of the fields (attributes). You can assume that there are no NULL values. All the fields of all the tuples have a value.
2. A user set. The set of users is nothing more than a set of ids. This is provided as a file, one user id per line. The file may be empty.
3. A query set, which is a set of queries that have been posed in the past. Each query has a unique query id and its definition (i.e., the set of conditions). The query set is expressed as a CSV file where each line contains the conditions attribute=value separated by comma. The first element of each row is a query id. For example, a row in the query set is:
   Q1821, name=John, age=55
4. A utility matrix User-Query that has for certain combinations of user and query, a value from 1 to 100 indicating how happy that user is with the answer set of that query. This is provided also as a CSV file, where the first row contains a list of query ids separated by comma. Any other row starts with a userId, followed by a set of comma separated fields (as many as the number of query names of the first row). These fields could be empty or have a value between 0 and 100. For example, the first row can be:
   q1, q4, q625, q81671
   and an example of one of the following rows
   user36182, 3,,88,3
   Note that the example above has one value missing (the one for the q4 (see the two consecutive commas)

The tool outputs the utility matrix received as input but with the missing values filled in with estimated values. That utility matrix can be kept in memory or exported in a CSV file. Given that matrix and a user u, one can return the top-k queries that may be of interest to the user u.

Part B:
Given what the part A has produced (i.e., the utility matrix), propose a way to compute the utility of a query in general for all the users. (recall that a query is actually a representation of a set of tuples). For instance the query city=Trento is a compact representation of all these tuples that have the value Trento in the city attribute.

You are asked to implement the above idea by thinking of a good way to implement it, describe it, and perform experiments that prove that you have developed a good solution.

The steps you need to perform are the following:

1. Describe the problem you are solving in general. This will be the introduction of your report.
2. Formally define the problem. This will be the problem definition part of the report.
3. Think of a solution and describe it in all its details (do not put any code. If you need to demonstrate something algorithmic, you do so with pseudocode. This is the solution section of your report.
4. Develop a dataset to try your solution. This dataset can be coming from the web or be generated by you automatically (synthetically)
5. Think and select a baseline with which you can compare your solution. It can be an existing algorithm or one of the solutions of your colleagues.
6. Do an experimental evaluation that provides a good understanding of all the different aspects of your solution and document your findings (section experimental evaluation of your report.

Deadline

20th December for the problem statement and the introduction (chapters 1 and 2 of your report)
8th January the Dataset
31st January Final report and code (the whole project)

Programming language

This is left totally to the project developers.

Number of persons

The project is for 2 persons.

Delivery

Create a Google drive folder called DM22_XXXXX where XXXXXX is your student number and share it with the instructor (velgias@unitn.it). Send an email to the prof with the link to that directory. If the project is of many people you can have in the naming the student number of all, e.g., DM22_XXXX_XXXX. In the folder you should place:

1. A pdf document called **report.pdf** structured as described in the section “Report Structure” below.
2. A directory called **doc** in which you will place the latex source files for the document report.pdf.
3. A directory called **src** in which you will place the code of your program. Include a README.txt file with instructions on how the program runs.
4. A directory called **data** in which you will place the data you have used in your experiments.

Evaluation Criteria

The project is evaluated according to the following evaluation criteria:

1. Novelty of the idea solution
2. Related Work understanding
3. Technical Depth: Detailed description of the approach and its challenging choices. How well the data mining algorithms you learned in the lecture have been used.
4. Presentation: Clarity and Completeness of the report
5. Research integrity. (Well defined formally the problem statement).
6. Experimental Evaluation
   - 6.1. The dataset(s) that have been used in the evaluation
   - 6.2. The evaluation tests that have been made (what has been tested and how)
   - 6.3. The comments on the results of the evaluation

Structure of the report

The final report should be written in Latex, using the following template http://velgias.github.io/tmp/template.zip It should contain the following sections:

1. **Introduction** (maximum 1 page) in which you introduce the problem you are solving, its importance and the main highlights of your solution (1 paragraph) and the results of your experiments (1 paragraph). Provide a

motivation for this work. (Why you think that such a study is important? And why it is challenging (i.e., not trivial) to perform this processing? What were the hard/challenging parts in developing a solution?) Note that a "hard/challenging" part should be generic and not personal to the authors. They should apply to everyone and are challenging due to the nature of the problem at hand. They should not be challenging just because of the capabilities of the author. For example, if the solution is developed in python and the programmer does not know python, then clearly the difficult is only for the specific author and not for everyone.

2. **Problem statement** (maximum 1 page) where you define formally what the problem you try to solve is.
3. **Related work** and technologies (maximum 1 page): Any information you think is important for the reader to know but IS NOT your own work. For example, you could describe there what k-mans is, what map reduce is, etc. Do not waste space for saying things that everyone else knows already from the lectures or other online sources. Keep it to the basic and to the minimum.
4. **Solution** (no max pages) In this section you describe in detail what your solution is. The more detailed you are in this section the better the section is. Imagine that you give your report to someone else and you ask her/him to implement your solution. Will that person be able to do it by looking only at what is written in the document? If yes, then the document is successful. In this part of the document, you should also include information about the function you have chosen for the quality measure. If you had other functions also in mind, explain why you did not choose them. You are free to include some pseudocode because it makes it much easier for people to understand what the text is saying.
5. **Experimental evaluation** (no max pages). This section contains a detailed description of all the experiments you have done to understand how well your solution works. How does it compare with the baseline? The more things you are testing, the more it helps to understand the performance of the solution, and the better the report is. The experimental evaluation should include both real and synthetic data. The size of the section is up to you, since it depends on the complexity of the solution you are proposing and the details you would like to study. Make sure that you also provide a description of the datasets you used as input.
6. **Conclusion**. A recap of what you did in your work (the main highlights). Maximum half a page.