(A glimpse on the course project so that you can start thinking of the solution. The full draft will be completed soon with the details on the final deliverable and the report structure).

On recognizing uniform areas in heterogeneous dataset. (A generalization of the clustering problem)

Data Intensive Systems Course, Utrecht University, 2021-2022

Instructor: Prof. Yannis Velegrakis

Course project. DEADLINE: July 4th, 2022

The goal of this work is to design, develop, implement, and test a method program that divides a dataset to a number of (maybe) overlapping subsets in such a way that each subset is as homogeneous[1] as possible.

A **dataset** is a collection of items. In this work we will consider as a dataset a relational table, and as items we will consider the records (meaning the tuples). The **homogeneity of a dataset** D is a metric *hom*(D) that indicates how homogeneous (similar) the elements of the dataset are. The homogeneity is the opposite of **heterogeneity** (this means that you can define heterogeneity as *1-homogeneity* (assuming that homogeneity takes values between 0 and 1), or you can define it as $\frac{1}{\text{homogeneity}}$).

A **decomposition** of a dataset D is a set of **k** datasets D_1, D_2, ..., D_k, such that $D_1 \cup D_2 \cup \ldots \cup D_k = D$, for i=1..k and each dataset D_i is a subset of D, i.e., $D_i \subseteq D$, for each i=1..k. We will call each subset D_i an **area.**

It is clear that when a dataset D is decomposed into smaller sets (the areas), the homogeneity of each subset (area) D_i can only be higher (or equal in the worst case) to the homogeneity that the original dataset D had. We are interested in finding a decomposition of a dataset D into areas D_1, D_2, ..., D_k such that the average of the homogeneities of these areas is maximized. In other words we are looking for a decomposition that maximizes the quantity *avg(hom(D_1), hom(D_2), ..., hom(D_k))*.

Note that the smaller the areas in a decomposition, the higher the homogeneity they have. As one can easily imagine, if we create subsets (areas) with one and only one item in each one, then the homogeneity is maximized. But such subsets (areas) are not useful. We would like to have meaningful areas which means that the number k of areas in which we divided the original set D should not be too large.[2]

It is known from Combinatorics that that the number of subsets of a set of N items is 2^N. The number of sets of such subsets is 2^{2^N} (in the worst case). This means that for a given dataset of N items, we can have 2^{2^N} possible decompositions. Not all these decompositions are interesting. We would like to restrict our attentions to only those that make sense. For this reason, we consider only decompositions that:

1. the union of their areas gives us the original set of N items (i.e., the fact that we mentioned earlier that $D_1 \cup D_2 \cup \ldots \cup D_k$ must be equal to D), and
2. every area D_i in the decomposition can be expressed in a declarative way as a conjunctive query, i.e., a query of the form $attr_1$=$value_1$ AND $attr_2$=$value_2$ AND … AND $attr_m$=$value_m$ .For example, the query firstname="John" AND age=10 AND city="Utrecht" describes the area of the relational table that consists of all the tuples that have the first name John, the age 10 and the city Utrecht.

[1] Homogeneous is the opposite of heterogeneous. Heterogenous means that there is a high variety.

[2] This is a situation similar to case of clustering. Given M items (points) one can always create one cluster for each different point but then this will not be of much use. There are techniques on how to decide right number of cluster (refer to the respective chapter 7 (section 7.3.3) in the book of Mining Massive Datasets http://www.mmds.org)).

You are asked to

1. Think and propose a good homogeneity function *hom(D)* of a dataset D
2. Implement a baseline solution for the decomposition problem. As baseline (standard solution) consider the traditional clustering (you can use k-means or any other clustering algorithm you like. Once you cluster the tuples, consider each cluster as an area, and compute its homogeneity. Take the average of the individual homogeneities to compute the homogeneity of the decomposition. Consider that value as the homogeneity of the baseline.
3. Devise a technique/algorithm to decompose a dataset D into a set of k areas such that the average homogeneity among the k areas is as high as possible for a specific value of k. (Describe how you determine a good value for an upper limit of k.) Your technique should run for very large relations, which means that it should exploit technologies for parallelization, namely spark or Map/Reduce. One solution is the exhaustive solution, i.e., the one that considers all the possible alternatives and selects the one that gives the better value, but other more advanced solutions are also welcome.
4. Perform experiments for datasets of different size and illustrate the time performance difference between the baseline and your approach. Explain why you believe the specific performance is achieved.

Deadline

4th of July for the delivery of the report and the code.

Dataset

The program is expected to accept a relational table in a CSV format, where every tuple is a line and the commas separate the different fields. It should work for any other dataset of this type. The goal of the work is to develop a generic system and not to make a study of a specific dataset. For this reason, a specific dataset is not provided. You should create your own test cases (synthetic data) but also try it with some real data and showcase the results that are produced. A good recommendation is the internet movie database (imdb) (https://www.imdb.com/interfaces).

Output - Visualization

It is up to you to decide the best way to present the output of your program and the the way to visualize it. This is not part of the assignment but having a nice presentation of the results makes the reader more comfortable.

Programming language

This is left totally to the project developers.

Number of persons

The project is for 2 or 3 persons.

Delivery

You need to deliver the code of the program you developed, the dataset you used, instructions on how the program runs and a report in which you describe the solution you have devised and the results of the experiments you have performed to prove the effectiveness and efficiency of your solution. To do that, you need to create a folder in your one-drive and share it (read and write permissions) with the instructor (i.velegrakis@uu.nl) and send the instructor a mail with the link. (Note that often just sharing is not enough so you need to also send the link). The folder should be called DIS22_XX_YY_ZZ, where XX, YY, and ZZ are the last names of the participants in the project.

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.
5. A directory called **results** in which you put the output of the runs of your program. The format of the file is up to you. The name of each file should start with the name of the dataset file that was used. For example, if the input file is movies.csv then the output file should be called movies_XXXXX where the XXXXXX is any informative string of your choice.

Presentation

On the last week of courses there will be a presentation in which a representative of every team will make a 5 min (maximum) presentation of the solution they are developing

Cross evaluation

After the delivery of the projects, you will also be asked to comment on the report of some other group (so make sure your report does not reveal your name, i.e, it is anonymous)

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. **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 Spark 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.
3. **Solution**. 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.
4. **Experimental evaluation**. 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.
5. **Conclusion**. A recap of what you did in your work (the main highlights). Maximum half a page.