On identifying the compensation for vaguely followed trips

Data Mining Course, University of Trento, 2023-2024

Instructor: Prof. Yannis Velegrakis

DEADLINE: January 15th, 2024 **Number of Persons: 2 or 3 or 4**

You are asked to solve some real-world problems. You have been hired by a logistics company that is looking for some technological solution to a situation it is facing. The company is responsible for transporting various forms of merchandise among different cities. The transport is done with trucks. A truck moves between two cities carrying one or more types of merchandise, each of a specific quantity. This kind of transportation of merchandise between two cities is referred to as a **trip**. For example,

<Milan, Verona, (milk:3, honey:20, butter:7, tomatoes:25)>

is an example of a trip from Milan to Verona, with 4 types of merchandise, of quantities 3, 20, 7, and 25, respectively. (The quantity is a number. It can be the number of items or Kgr). A truck may perform a sequence of trips, where the end of one trip is the beginning of the next. This sequence of trips is known as a **route**. An example of a route with 3 trips is the following:

<Rome, Milan, (milk:3, pens:10, butter:20)>
< Milan, Verona, (milk:5, honey:9, butter:10, tomatoes:20)>
< Verona, Venezia, (butter:7, pens:2, tomatoes:10)>

which starts from Rome and through Milan and Verona arrives in Venezia.

The logistics company, based on the orders it often receives, has used a special software that has made some suggested routes, i.e., sequence of cities with the respective merchandise type and quantity in each trip. These are referred to as the **standard routes**. Every time the company needs to implement one of the standard routes, it hires a truck (with its driver of course) and asks the driver to perform the route. At the end of the route the truck driver sends to the company the route it followed, i.e., the ordered sequence of trips, and requests payment. The sending of the route that the track followed is done automatically because the truck is equipped with some special hardware (NFC) and also with GPS, so all the necessary route information is composed automatically.

All seems easy and clear, but reality is not, and this is where the challenges start. Drivers never stick exactly to the plan, meaning the standard route that they have been hired to do. They often load a little more or a little less quantity of one or more merchandise. Other times, they may load some extra merchandise or omit some. Last, but not least, they may add some extra city (or cities) in their route, or they may omit some. All this means that the actual route that a truck has executed, is very rarely the standard route the truck driver was asked to execute. We will refer to these routes as the **actual routes.** The reason they do that is because they have some personal agenda or preferences. When the track sends the information to the company at the end of the route, since the information is sent automatically, it is the actual route and not the standard route that was sent to the driver.

This is clearly a situation that the company does not like and would like to minimize the discrepancy between what it asks the drivers to do (the standard routes) and what the drivers actually do (the actual routes). They thought of two ways to do this. The first is to assign to drivers the routes that are as close as possible to their hidden agenda and preferences, and the second is to update the standard routes that the company has to those that match as much as possible the hidden agenda and preferences of the drivers as a whole. Given this you have been hired to devise a system that, given the set of standard routes that the company has now and the actual routes that the drivers have done so far, it produces

i) a recommendation to the company on what standard routes it should have,
ii) for each driver, creates a list of standard routes in that order so that the higher in the list a standard route is, the least the diversion of the driver will be, and
iii) for each driver, generates the ideal standard route that the driver will have the least divergence.

Programming language

This is left totally to you.

Number of persons

The project is for 3 or 4 persons.

Input Dataset

There is no specific data that is given to you. This is for privacy reasons. The company cannot share with an external person the internal company data. When you develop the program, and assuming that you make it in such a way that it works well, the company will make it run on their data and see the results. This means that you have to create your own datasets and test cases to make sure that it works well and prove it (this is called synthetic Dataset Generation). For instance, you can randomly create a set of standard routes and then have a piece of code that given the standard routes, for each standard route you create a variation and then with this variation you create an actual route. You need to create a large dataset (do not create toy examples). You need a lot of standard routes and very many actual routes.

The input to your program is a JSON file called standard.json that contains the actual routes. The syntax of the JSON file is the one you see below, which is believed to be self-explanatory. Note that every route also has an identifier so that we can refer to it if needed.

```
[
        {id:s5, route:[
                        {from:'Rome', to:'Milan', merchandise: {milk:3, pens:10, butter:20}},
                        {from:'Milan', to:'Verona', merchandise: {milk:5, honey:9, butter:10, tomatoes:20}},
                        {from:'Verona', to:'Venezia', merchandise:{butter:7, pens:2, tomatoes:10}}
                ]
        },
        {id:s10, route:[
                        {from:'Rome', to:'Milan', merchandise: {milk:2, pens:10, butter:20}},
                        {from:'Milan', to:'Verona', merchandise: {milk:5, tomatoes:24}},
                        {from:'Verona', to:'Venezia', merchandise:{butter:7, bread:2, tomatoes:10}}
                ]
        },

]
```

There is also a second file as input called actual.json that contains the standard routes. The syntax is exactly as above but it has the additional information of who is the driver that implemented the route, and what standard route the driver had been asked to implement. For example:

```
[
        {id:a25, driver:C, sroute:s5, route:[
                        {from:'Rome', to:'Milan', merchandise: {milk:4, pens:4, butter:20}},
                        {from:'Milan', to:'Bergamo', merchandise: {milk:5, honey:19, butter:10, tomatoes:20}},
                        {from:'Bergamo', to:'Venezia', merchandise:{butter:47, pens:2, tomatoes:1}}
                ]
        },
        {id:a13, driver:E, sroute:s10, route:[
                        {from:'Bolzano', to:'Milan', merchandise: {milk:2, pens:10, butter:22}},
                        {from:'Milan', to:'Verona', merchandise: {milk:15, tomatoes:4}},
                        {from:'Verona', to:'Venezia', merchandise:{butter:7, bread:21, tomatoes:10}}
                ]
        },

]
```

Output

The output of the program is:

1. a file called recStandard.json that has a syntax exactly as the standard.json and contains your recommendation to the company on what standard routes it should have.
2. a file called driver.json that has for each driver, the 5 standard routes routes that if the driver does them, it minimizes the diversion. You can test this by considering as pool of standard routes those that originally the company has and also those that you recommend in the recStandard.json. The file driver.json has the following syntax:

```
[
        {driver:C, routes:[s10, s20, s2, s6, s10]},
        {driver:A, routes:[s1, s2, s22, s61, s102]},
        ….

    ]
```

3. a file called perfectRoute.json that contains for each driver what would be the perfect ideal route. The format of the file is as follows:

```
[
    {driver:C, route:[
            {from:'Rome', to:'Milan', merchandise: {milk:4, pens:4, butter:20}},
            {from:'Milan', to:'Bergamo', merchandise: {milk:5, honey:19, butter:10, tomatoes:20}},
            {from:'Bergamo', to:'Venezia', merchandise:{butter:47, pens:2, tomatoes:1}}
        ]
    },
    {driver:E, route:[
            {from:'Bolzano', to:'Trento', merchandise: {milk:2, pens:10, butter:22}},
            {from:'Trento', to:'Verona', merchandise: {milk:15, tomatoes:4}},
            {from:'Verona', to:'Venezia', merchandise:{butter:7, coca-cola:21, tomatoes:10}}
        ]
    },

]
```

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 google-drive and share it (**read and write permissi**ons) with the instructor (velgias@unitn.it) and **send the instructor a mail with the link.** (Note that just sharing is not enough so you need to also send the email with the link). The folder should be called DM23_XX_YY_ZZ, where XX, YY, and ZZ are the last names of the participants in the project. It should contain:

1. A pdf document called **UUUU.pdf** where UUUU is the name of your team and should be structured as described in the section "Report Structure" below.
2. 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.
3. A directory called **data** in which you will place the data you have used in your experiments.
4. A directory called **results** in which you put the output of the runs of your program. The name of each file should end with the name of the dataset file that was used. For example, if the input file is actual1.json then the output file should be called recStandard1.json.

Evaluation Criteria

The project is evaluated according to the following evaluation criteria:

1. Novelty & sophistication of the idea as well as how well it solves the problem.
2. Technical Depth: Detailed description of the approach and its challenging choices. How well data mining solutions have been exploited.
3. Presentation: Clarity and Completeness of the report & the presentation
4. Experimental Evaluation
    4.1. The dataset(s) that have been used in the evaluation
    4.2. The evaluation tests that have been made (what has been tested and how)
    4.3. The comments on the results of the evaluation

Structure of the report

The final report should be written in Latex, using the following template which is available on github: 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 motivation for this work. (Why do you think that such a study is important? (you already have an application so it is clear that is important, but maybe you can think of additional applications to make the statement stronger). Why is it 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 difficulty 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 what k-means is, what collaborative filtering is, etc. Do not waste space getting into details that everyone else knows already from the lectures or other online sources. Keep it to the basics 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. Also explain the reasoning behind every choice you are making. 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 to some baseline? The more things you are testing, the more it helps to understand the performance of the solution, and the better the report is. Since the company refuses to share its datasets, the experiments will be on 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.