Algo 2 matching, graphs, clustering : project

NICOLAS LE HIR
nicolaslehir@gmail.com

1 DESCRIPTION OF THE PROJECT

The goal of the project is too analyze and process a dataset of your choice, using methods studied during the course.

1.1 Dataset constraints

You are free to choose the dataset within the following constraints :

- utf-8 encoded in a **data.csv** file
- several hundreds of lines
- around 10 attributes (columns), the first being a unique `id`, separated by commas
- some fields must be quantitative (numbers), others categorical (not numbers).
- some fields could be correlated (for instance there is a correlation between temperature and pressure as seen in class)

It's nice if the dataset comes from a real example but you can also generate it.
Example datasets available :
`https://github.com/awesomedata/awesome-public-datasets`
`https://www.kaggle.com/datasets`
Note for students who attended the Visualization module : please use a different dataset than the one you worked with if you did the project.

1.2 Processing

The processing must be made with **python** (**python** 3 **preferred**) with **two files** :

build_graph.py Generates of a graph to describe the compatibilities between the datapoints (build edges between some of them).

You have to choose how to build the compatibility graph : as we have seen in class, there might be several relevant ways to do it.

However, most probably you will have to :

- 1) quantify the non-quantitatives variables,
- 2) normalize and/or weight the importance of the different fields,
- 3) remove useless variables,
- 4) create a distance or a similarity between datapoints,
- 5) set a threshold and

— 6) build edges between points that are seperated by a distance smaller than the threshold, or between which the similarity is higher than the threshold.

match.py or **cluster.py** Returns either

— a maximum matching in the created graph (extration of the greatest possible number of pairs of compatible elements)

— OR, more relevant, groups of datapoints containing a number $\geqslant 3$ of elements (ie a cluster).

In the case of the matching in a bipartite graph, we have seen in the course that there exists a polynomial algorithm to exactly solve the problem.

In the case of a non-bipartite graph, other algorithms exist, such as the blossom algorithm. You may try it or implement an heuristic of your choice.

In the case of the clustering, the problem might require a harder algorithm in terms of complexity. You can thus use a heuristic to solve it, and/or a classical method.

The processing must return a **result.csv** file containing a list of pairs (for a matching) or groups (for a clustering) of ids representing the matching or the clustering.

Important :

The usage of this dataset and its processing should be justified by a question of your choice. Thus, the approach should be explained and justified in a separate pdf file. The pdf file needs to contain explanations about :

— the nature of the dataset

— information on the potential correlation between variables.

— justifications about its processing, and in particular the construction of the distance or similarity.

— description of the matching or clustering used.

— comments on the results obtained.

The more interesting and original the dataset is, and the more relevant and justified the construction and matching in the graph, the higher the score.

Don't hesitate to use networx or graphviz or another tool to illustrate the graph created, or parts of the graph.

Example references include [**?**].

2 ORGANIZATION

The students can form groups with at most 3 students (they can work alone too).

The deadline for submitting the project is **March 8th 2020**. You may send compressed folder or a repo containing :

— the name of the student(s)

— the csv dataset **data.csv**

— the algorithm **build_graph.py**

— the algorithm **match**

Please write "Algo 2 matching session 3" in the subject of your email.

You can reach me by email, I will answer faster if you use the gmail address rather than the epitech address.

3 EXERCISES DONE DURING THE COURSE

The exercises we made during the class are available with correction here : https://github.com/nlehir/ALGO2. The repository contains example functions you can use to create graph images, such as **random_graph.py**. It is not mandatory to use this repo.

4 LIBRARIES

You may use third-party libraries : however, if you do so, it is required that you present them in your document and describe the functions that you use from thie library, and comment on the choice of the parameters.

5 VALIDATION

The project is not mandatory and can offer two extra credits, apart from the credit based on being present in the class.