Algorithmic complexity and graphs: compatibility graphs

1er octobre 2022

Compatibility graphs

- Yesterday we processed graphs describing **relationship between data**
- If two nodes were related, they were linked by an edge in the graph.

Compatibility graphs

- Yesterday we processed graphs describing **relationship between data**
- If two nodes were related, they were linked by an edge in the graph.
- Today we are interested in building such graphs directly from the data, we call them **compatibility graphs**.

Compatibility graphs

We are interested in building **compatibility graphs**.
Given two nodes in a graph, should there be an edge between them ?

Compatibility graphs

We are interested in building **compatibility graphs**.
Given two nodes in a graph, should there be an edge between them ?
Note : it is not the same problem as the matching problem. In the matching problem, the edges are already defined.

Compatibility graphs

We are interested in building **compatibility graphs**.
Given two nodes in a graph, should there be an edge between them ?
Note : it is not the same problem as the matching problem. In the matching problem, the edges are already defined.
However, once the edges are built, we can apply a matching to it.

Example applications

- Social networks management
- Recommendations

Building compatibility graphs

- We will build graphs first from simple data
- Then from more complex data.

Building a graph from simple data

- We will first build a graph from simple data in the 2D space.

Euclidian distance and compatibility

Consider the following data :

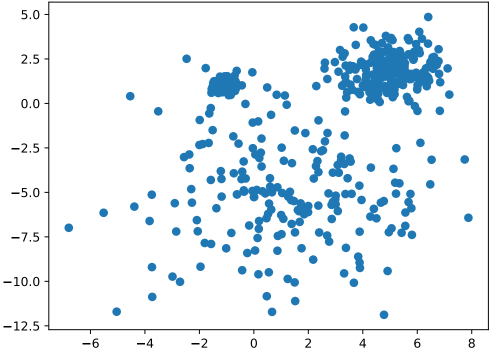


Figure – Data : we would like to define **edge** between some of them

Is this set of edges a good solution ?

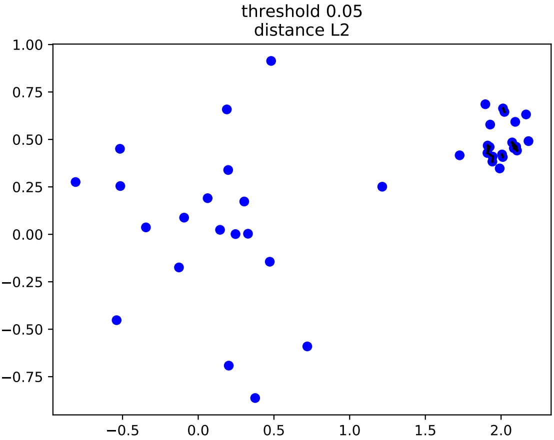


Figure – Some definition of edges

Is this set of edges a good solution ?

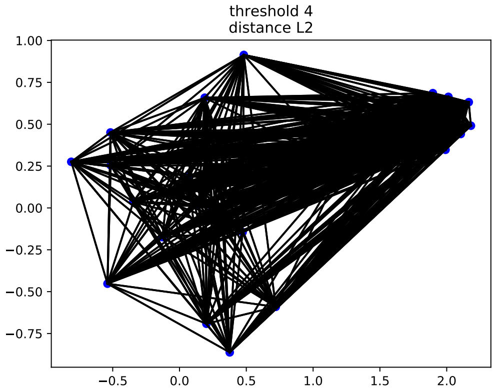


Figure – Some definition of edges

This one looks ok

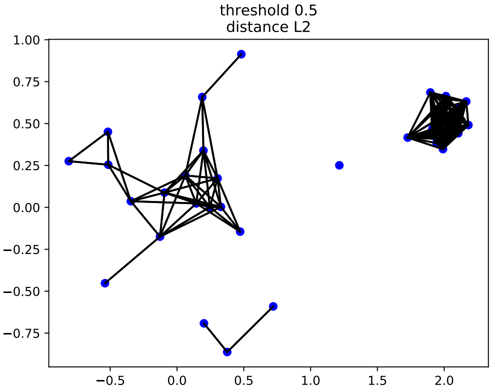


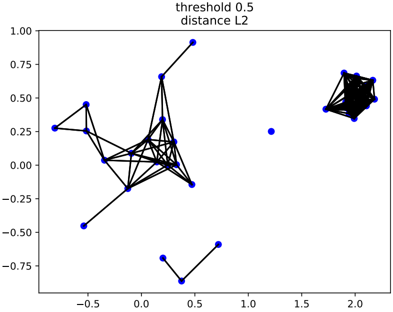
Figure – A proposition of edges

Backboard

- Euclidian distance and threshold.

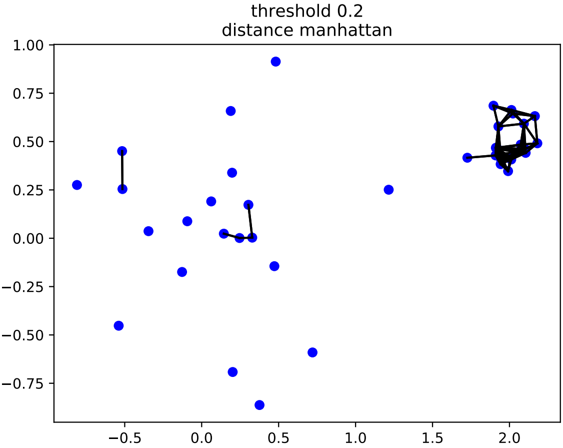
Exercice 1 : Setting a threshold

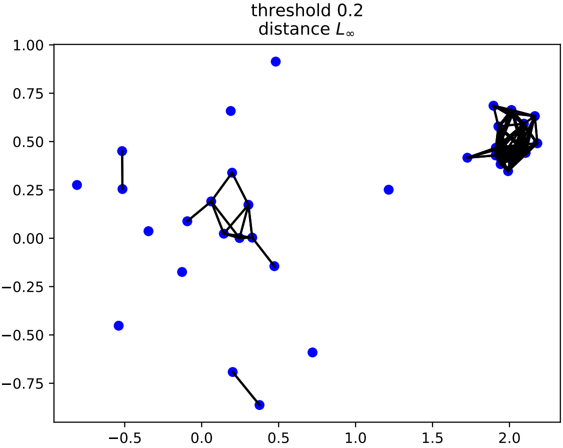
cd compatibility_graphs/geometric_data and set the threshold used in **build_graph_1.ipynb** to draw relevant edges between the nodes. Feel free to use another dataset !

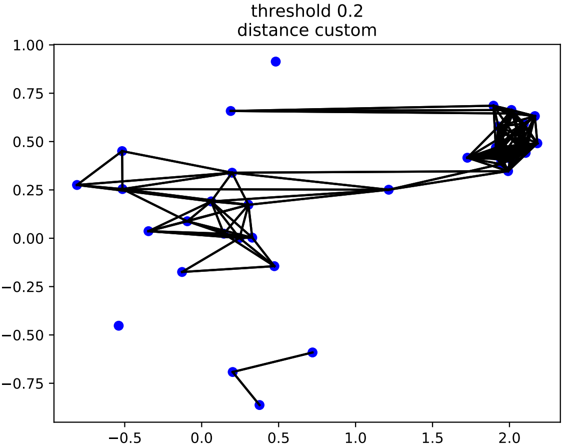


Exercice 2 : Changing the distance

- Assess the impact of changing the distance used. Possible choices :
  - $L1$ distance (Manhattan)
  - $|| ||_\infty$ distance (backboard)
  - custom distance
- use **build_graph_2.py** and edit the distances used at the end of the file.
- Try several values for the threshold.


threshold 0.2
distance manhattan
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−0.5
0.0
0.5
1.0
1.5
2.0


threshold 0.2
distance L_∞
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−0.5
0.0
0.5
1.0
1.5
2.0


threshold 0.2
distance custom
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−0.5
0.0
0.5
1.0
1.5
2.0

General notion of a distance

- Let us generalize what we experimentally studied.

Examples of distances

$x = (x_1, \ldots, x_p)$ and $y = (y_1, \ldots, y_p)$ are p-dimensional **vectors**.

Examples of distances

$x = (x_1, ..., x_p)$ and $y = (y_1, ..., y_p)$ are p-dimensional **vectors**.

- $L_2 : ||x - y||_2 = \sqrt{\sum_{k=1}^{p} (x_k - y_k)^2}$ (Euclidian distance, 2-norm distance)

Examples of distances

$x = (x_1, \dots, x_p)$ and $y = (y_1, \dots, y_p)$ are p-dimensional **vectors**.

- $L_2 : ||x - y||_2 = \sqrt{\sum_{k=1}^{p} (x_k - y_k)^2}$ (Euclidian distance, 2-norm distance)
- $L_1 : ||x - y||_1 = \sum_{k=1}^{p} |x_k - y_k|$ (Manhattan distance, 1-norm distance)

Examples of distances

$x = (x_1, ..., x_p)$ and $y = (y_1, ..., y_p)$ are p-dimensional **vectors**.

- $L_2 : ||x - y||_2 = \sqrt{\sum_{k=1}^{p} (x_k - y_k)^2}$ (Euclidian distance, 2-norm distance)
- $L_1 : ||x - y||_1 = \sum_{k=1}^{p} |x_k - y_k|$ (Manhattan distance, 1-norm distance)
- weighted $L_1 : \sum_{k=1}^{p} w_k |x_k - y_k|$

Examples of distances

$x = (x_1, ..., x_p)$ and $y = (y_1, ..., y_p)$ are p-dimensional **vectors**.

- $L2 : ||x - y||_2 = \sqrt{\sum_{k=1}^{p} (x_k - y_k)^2}$ (Euclidian distance, 2-norm distance)
- $L1 : ||x - y||_1 = \sum_{k=1}^{p} |x_k - y_k|$ (Manhattan distance, 1-norm distance)
- weighted $L_1 : \sum_{k=1}^{p} w_k |x_k - y_k|$
- $L_\infty : \max(x_1, \ldots, x_n)$ (infinity norm distance)

Hamming distance

- $\#\{x_i \neq y_i\}$ (Hamming distance)

Hamming distance and edit distance

- $\#\{x_i \neq y_i\}$ (Hamming distance)
- linked to **edit distance** : used to quantify how dissimilar two strings are by counting the number of operations needed to transform one into the other (several variants exist)

General definition of a distance

A **distance** on a set E is an application $d : E \times E \to \mathbb{R}_+$ that must :

General definition of a distance

A **distance** on a set E is an application $d : E \times E \to \mathbb{R}_+$ that must :

- be **symmetrical** : $\forall (x, y) \in E^2, d(x, y) = d(y, x)$

General definition of a distance

A **distance** on a set E is an application $d : E \times E \to \mathbb{R}_+$ that must :

- be **symmetrical** : $\forall (x, y) \in E^2, d(x, y) = d(y, x)$
- **separate the values** : $\forall (x, y) \in E^2, d(x, y) = 0 \Leftrightarrow x = y$

General definition of a distance

A **distance** on a set E is an application $d : E \times E \to \mathbb{R}_+$ that must :

- be **symmetrical** : $\forall (x, y) \in E^2, d(x, y) = d(y, x)$
- **separate the values** : $\forall (x, y) \in E^2, d(x, y) = 0 \Leftrightarrow x = y$
- respect the **triangular inequality**
  $\forall (x, y, z) \in E^3, d(x, y) \leq d(x, z) + d(z, y)$

Building compatibility graphs for non geometrical data

- Some data are not geometric
- Some features are not numbers, but could for instance be strings, or categories (categorical data)
- We will use **pandas** process the data from a **csv** file and build a compatibility graph.
- You can use **compatibility_graphs_other_data/build_graph.py** or the notebook.

Non geometrical data

Exercice 3 : **Experiment with the data, the threshold in order to build different graphs**