Behavioral Pattern Cluster Analysis System

POLITECNICO
MILANO 1863

ZHANG YUE DONG

****

PREFACE

Student : Zhang Yuedong

Matricola : *********

Codice Persona : *********

E-mail : **********

Professor : Dr. Fabio Salice

Assistant teacher : Dr. Andrea Masciadri

Index

Introduction

This is a python program to execute cluster analysis for people's daily living behavior at home, it used agglomerative hierarchical clustering to cluster which days are similar and present the common pattern for similar days.

We named this program as “Behavioral Pattern Cluster Analysis System (BPCAS)”, For user, it’s very user-friendly cause it has a Command-line Interface (CLI), this CLI will guide the user to execute the right input and present the program running log. For operator, the code is very readable and maintainable.

This documentation has two parts, Usage and Architecture of BPCAS.

For the first part, the usage is a guidebook for “getting started”, we recommend that you read this section carefully before using, especially the dataset part, because BPCAS is designed for the special dataset, “ensuring that the dataset is valid” is the key to the correct operation of this program.

For the second part, if you only care about how to use it and don’t care about the internal implementation, you can ignore this part directly. This part is the architecture of BPCAS, it details the architecture of the program, from execution process to algorithm details, from data extraction to data presentation, they will be illustrated and explained as clearly & simple as possible.

So, here we go.

Usage

The source code is under the path src/main, first of all, build the third-party libraries, and then run “ __init__.py ” file with python 3.6. The src/test is unit test code.

2.1 Command-line Interface (CLI)

The CLI is very user-friendly, you just have to insert the values following the instructions, and in case of an error, you'll be notified.

```
Welcome to behavioral pattern cluster analysis system!
Enter the dbname:
Enter the database user:
Enter the password(If there is no password, please press enter):
Enter the host(press enter for the default value):
Enter the port(press enter for the default port 5432):
Connection database succeeded!
There are 4 kinds of sensor data, which one do you want to analyze?
1.the PIR sensor
2.the lumen sensor
3.the temperature sensor
4.the power sensor
5.all of them
Enter: 5
Building PIR sensor list...
Building Lumen sensor list...
Warning: Room 9 No lumen signal received at 2020-04-01
Warning: Room 10 No lumen signal received at 2020-04-01
```

```
Preparing exec hierarchical clustering
please enter the the desired number of clusters(max cluster) for presentation the common pattern( recommended 1 to 5 ).
Enter: 4
Executing hierarchical clustering...
Please enter the level of critical distance(DT), you can press Enter to input the default value(0.7*max(Z[:,2])).
Hint: The corresponding level of critical distance of the desired number of clusters is 6842.6
Enter: 6842.6
Executing dendrogram presentation...
Executing calendar presentation...
Executing common pattern presentation...
Presenting abnormal days: 2020-04-08
Presenting common pattern: 2020-04-06  2020-04-07
Presenting common pattern: 2020-04-03  2020-04-04  2020-04-05
Presenting common pattern: 2020-04-01  2020-04-02
Hint:Maximize the window to get the best visual effect.
```

2.2 Input

2.2.1 Dataset

Connection the database

There are 5 arguments.

- dbname: compulsory, the name of database.
- database user: compulsory, the database administrator.
- password: optional, the database password, if there is no password, just press Enter.
- host: optional, the database host, press Enter for “localhost”
- port: optional, the database port, press Enter for default port 5432

If the input fields have some problems, the program will exit automatically, please check the connection information and run it again.

Check if the dataset is applicable

Three tables are necessaries.

- person_position: This table stored the PIR sensor’s signals, data will be extracted from id_room, t_from, t_to attributes.
- sensor: This table stored configuration information of all sensors, data will be extracted from id_room, id_sensor, id_sensor_type, threshold attributes.
- stream_data: This table stored the Lumen Temperature and Power sensor’s signals, data will be extracted from value, id_sensor, timestamp attributes.

Determine the id of these three sensors in “sensor_type” table like this

ID	NAME
3	Light
4	Temperature
5	Energy

Determine the appliances’ id in “sensor” table like this

ID_SENSOR	NAME
24	Microonde
26	Televisione
28	HC2 Power
32	Frigorifero
34	Forno
36	Lavatrici
45	Serra A
148	Lavastoviglie
150	PC

Dataset Error & Warning

-Error

In case of a failed access to the database, an error message will be showed, and the program will exit automatically.

```
Welcome to behavioral pattern cluster analysis system!
Enter the dbname: test
Enter the database user: test
Enter the password(If there is no password, please press enter):
Enter the host(press enter for the default value):
Enter the port(press enter for the default port 5432):
 Connection database error.  FATAL:  role "test" does not exist

 Program EXIT, please check the database problem and reopen it, think you!

Process finished with exit code 0
```

-Warning

Warning message will not cause the program to exit. There are two cases of warning

- Warning for appliance. In the dataset, there are 9 appliances. If other appliances appear, this warning will raise, you can ignore it if you don't care the new appliance.
- Warning for signals: This warning will raise if some sensors have no signal throughout the day.

For example:

Warning: Room 9 No lumen signal received at 2020-04-01
Warning: Room 10 No lumen signal received at 2020-04-01
Warning: Room 9 No lumen signal received at 2020-04-02
Warning: Room 10 No lumen signal received at 2020-04-02

This means that in the entire day of 2020-04-01, there is no lumen signal in the ninth room, maybe the lumen sensor was not installed that day. Anyway, if you are sure that the dataset is correct, you can ignore it.

2.2.2 The desired number of clusters

This value also called max_cluster, it will decide the number of common patterns to present in the output images.

2.2.3 The level of critical distance(DT)

This value is the height of the critical distance, it will decide the final cluster in the dendrogram, the default value is 0.7*max (max distance of all cluster). Of course, we recommend using the corresponding value of max_cluater, and when you type, CLI will prompt you what the value is.

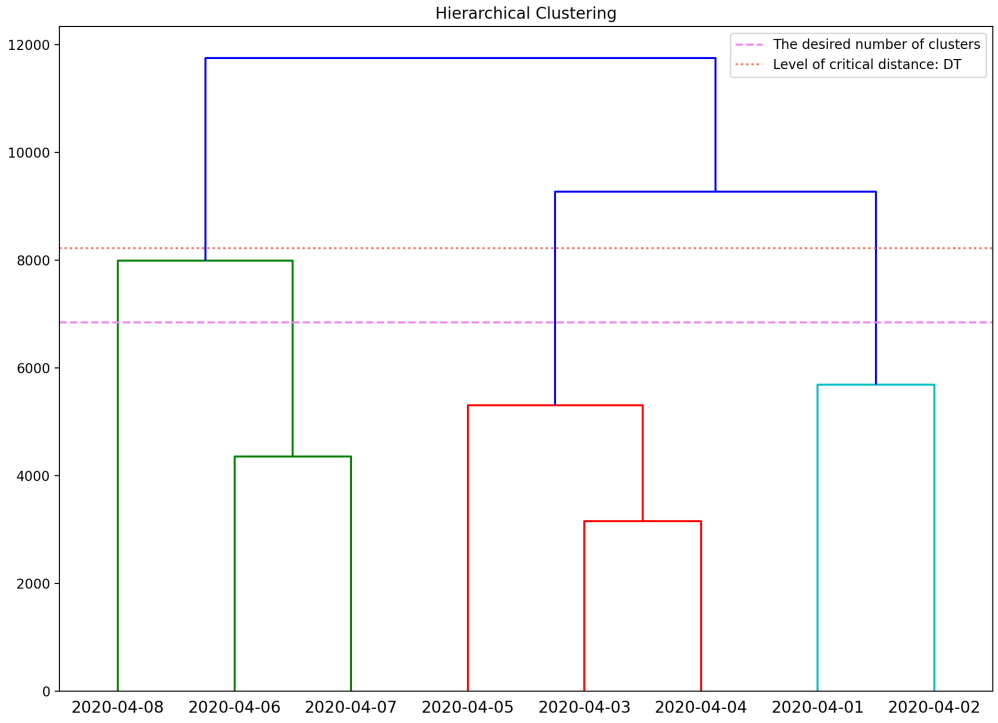
For example:

```
Preparing exec hierarchical clustering
please enter the the desired number of clusters(max cluster) for presentation the common pattern( recommended 1 to 5 ).
Enter: 4
Executing hierarchical clustering...
Please enter the level of critical distance(DT), you can press Enter to input the default value(0.7*max(Z[:,2])).
Hint: The corresponding level of critical distance of the desired number of clusters is 6842.6
Enter: 6842.6
```

2.3 Output

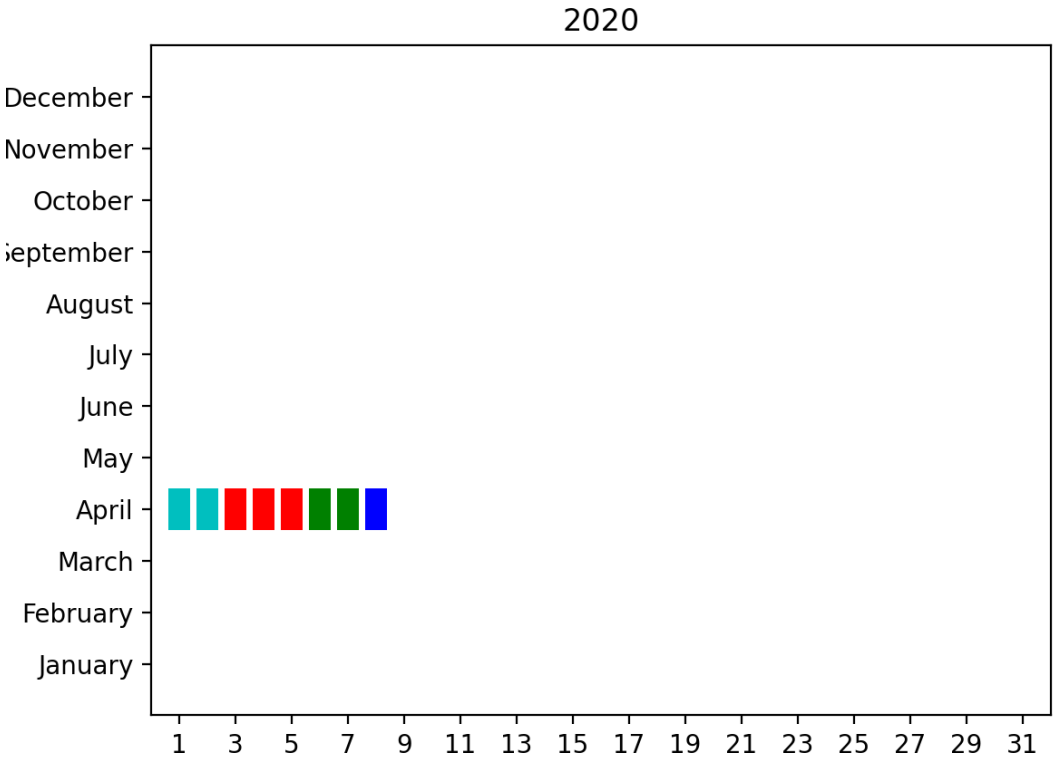
2.3.1 Dendrogram

In the dendrogram image, except the dendrogram, it also will present max_cluster and the level of critical distance(DT).
All days below the level of critical distance(DT) will be treated as the same cluster, and the desired number of clusters (max cluster) will present those clusters it passed.
For example:



As mentioned above, DT decides to dendrogram, and max_cluster decide the number of common patterns or abnormal day, if these two values are in the same interval, the common pattern and abnormal day will perfectly present the content of dendrogram.

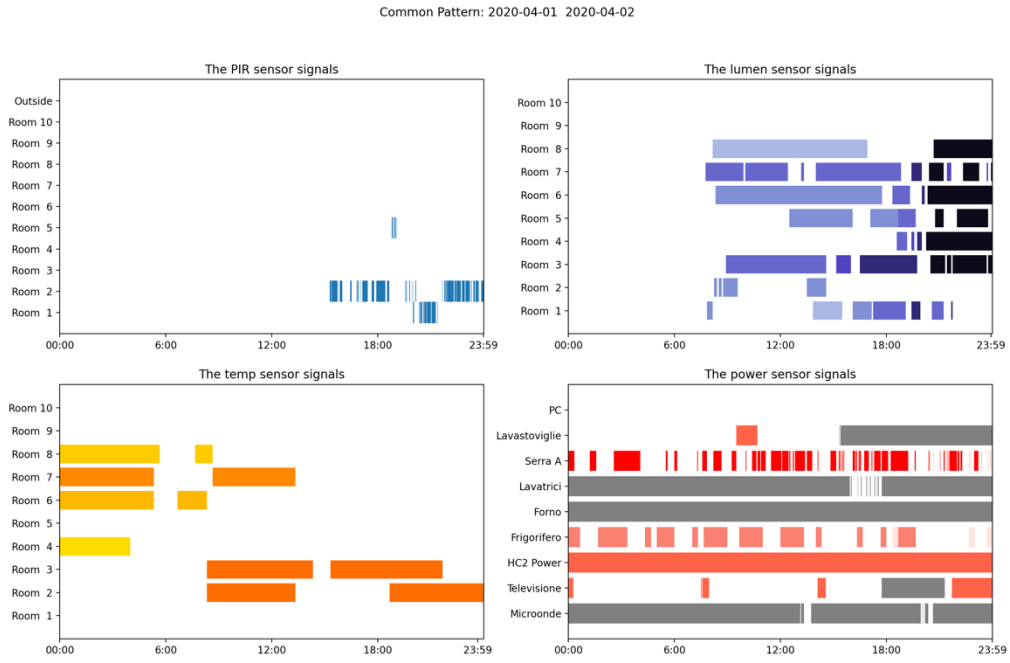
2.3.2 Calendar

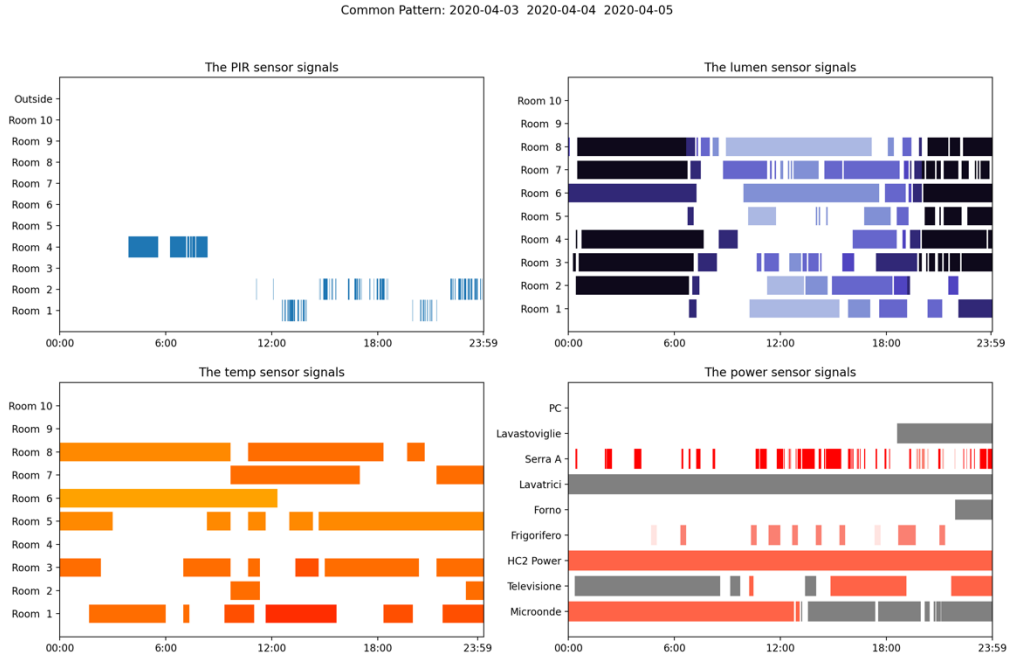


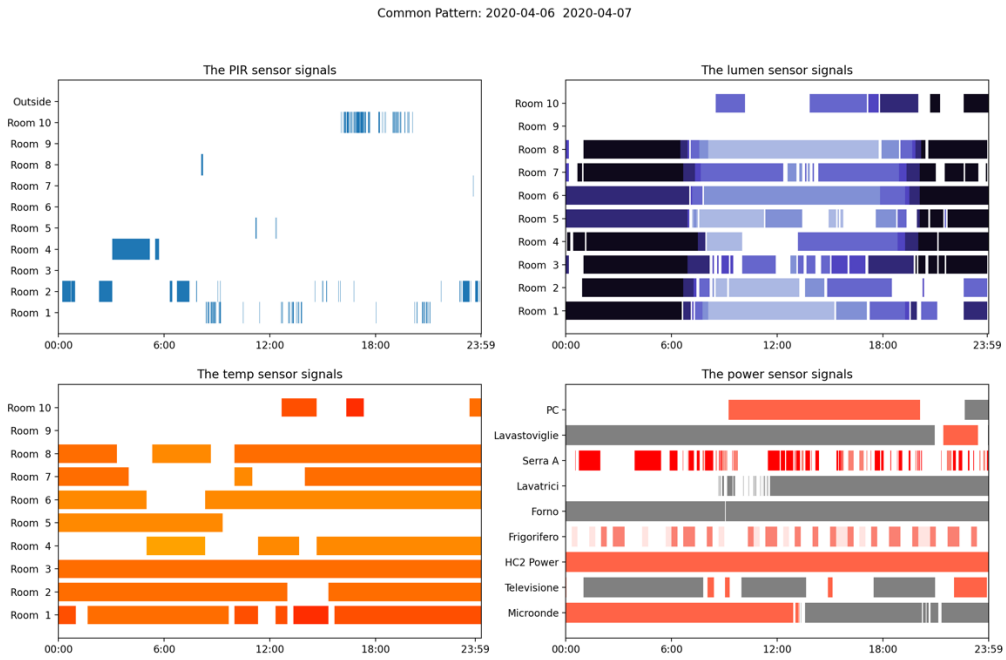
This figure will present which days in the same cluster, their colors are the same as in the dendrogram, the abnormal days will be colored blue.

2.3.3 Common pattern

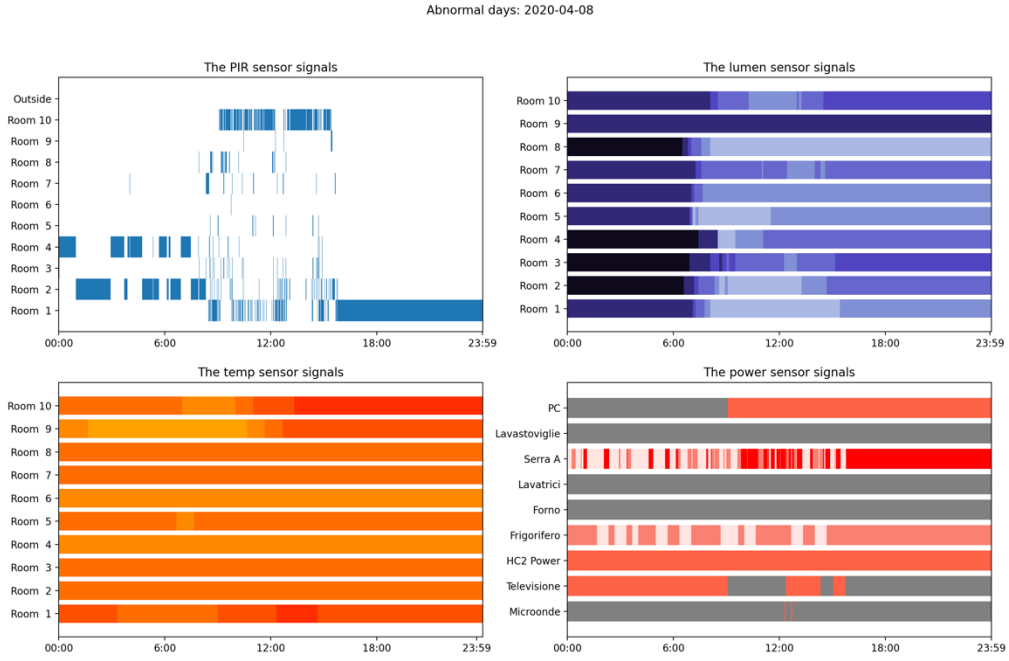
For lumen signals, the darker the color, the darker the light and vice versa.
For temperature signals, the redder color means higher temperature and vice versa.
For power signals, the red color means on, the gray means off. But for “Serra A” and “Frigorifero” the redder color means larger power and vice versa.







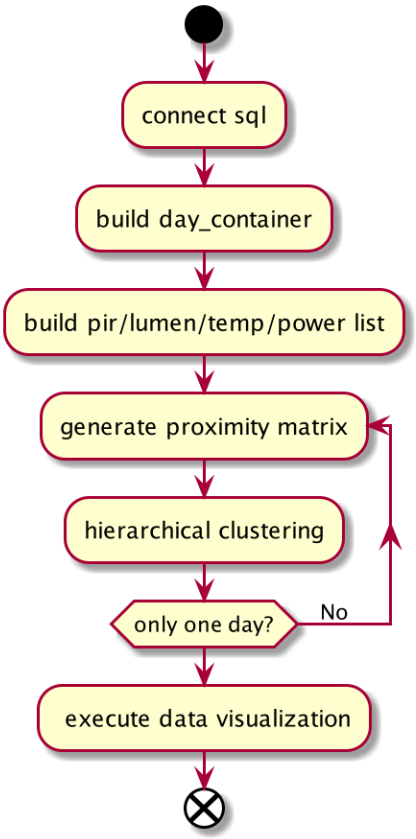
2.3.4 Abnormal day



Program Architecture

3.1 Dataset Connection

We used the three-party library “psycopg2” to handle database connection. About this section, please refer to code class SQLTool.



3.2 Build day container

In this process, all dates have to be extracted from the dataset with

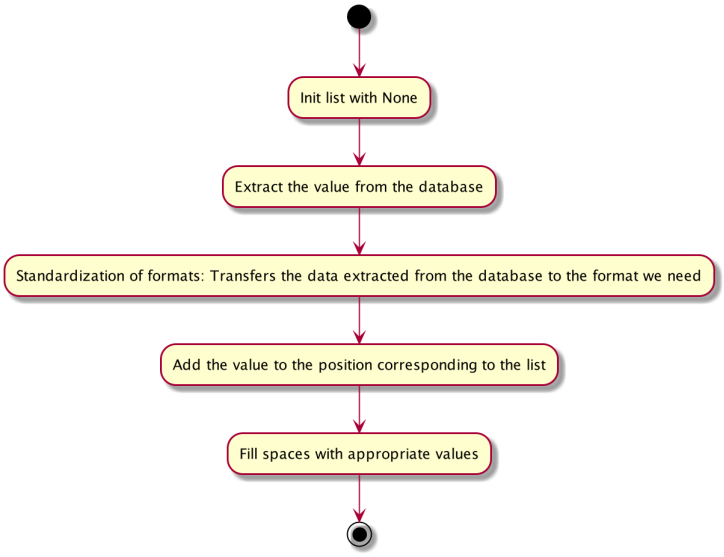
```
SELECT date(t_from) as date FROM person_position group by date order by date;
```

query statement.
Then build a class day_container for each day, store them in the class day_deck.

3.3 Build sensor list

The build sensor list process simulates the ETL (Extraction, Transformation e Loading) operation set, for each type of sensor list, we have to extract information from the dataset with a query statement and transform the formats according to the characteristics of each sensor. Finally, load them into the corresponding position. For every kind of sensor, we have a concept named “sampling interval”, it means that the interval time we need for each sampling. For all lists, we initialize them to “None”.

Process:



3.3.1 Build PIR sensor list

- List format : 1-dimension list, length 2880.
- Sampling interval : 30s once, 2880 times per day.
- Database query statement :

```
SELECT id_room, t_from, t_to FROM person_position  WHERE date(t_from)= + date_curr +  or date(t_to)= + date_curr
+ order by t_from, t_to, id_room;
```

- Standardization of formats : The data type of PIR value is string, ‘1’ ~ ‘9’ for room 1 ~ 9, ‘a’ for room 10, ‘f’ for not at home (that is outside).
- Overlap problem : If multiple sensors are active at the same time, this means that the person is moving in multiple rooms, we handle this case like the example : If this person moves from room 4 and pass room 3 to room 5, we record string ‘435’.
- Fill black spaces : When all the values from the dataset were been inserted, some parts may still blank, so, they have to be filled with an appropriate value. The last char of the previous signal will use to be filler, but if the first signal is black, just fill with the first char of the following "not black signal". As shown on the below image, the PIR sensor does not continuously record the location of people, so all initial values should be filled with appropriate values.

```
For example: previous is '34', we just use '4',
Because signal 4 is the last presented.
 And
 [None,None,'34','4','43',None,None,'54','4,.......]
 -> ['3','3','34','4','43','3','3','54','4',.......]
```

3.3.2 Build Lumen sensor list

- List format: 2-dimension list, length [10][288], 10 rows for each room, 288 columns for each signal.
- Sampling interval: 300s once, 288 times per room per day.
- Database query statement:

```
SELECT se.id_room, sd.value, sd.timestamp FROM sensor as se  join stream_data as sd on se.id_sensor = sd.id_sensor
WHERE se.id_sensor_type = 3 and date(timestamp) =  + date_curr + order by se.id_room, timestamp;
```

- Standardization of formats: we have to transform the signal value to lumen level and determine the lumen level according to sunrise and sunset, the sunrise and sunset times of Milan in April are from 6:30 ~ 20:00. So, lumen levels are from 0 to 5 (which are from 'buio' to 'ottima')

  In table:

Sera			**Giorno**	
luminosità	Valutazione		luminosità	Valutazione
= 0	BUIO		= 0	BUIO
0 < x <= 5	SCARSA		0 < x <= 20	SCARSA
5 < x <= 20	DISCRETA		20 < x <= 40	DISCRETA
2 0< x <= 110	BUONA		40 < x <= 150	BUONA
110 < x <= 200	MOLTO BUONA		150 < x <= 300	MOLTO BUONA
200 < x	OTTIMA		300 < x	OTTIMA

- Fill black space: Same as the PIR sensor list, the black spaces have to be filled, the previous signal will use to be filler, and if the first signal is black, fill with the following "not black signal". For example, we will fill list from [None, None, '3', '4', '3', None, None, '5', '4, ...] to ['3', '3', '3', '4', '3', '3', '3', '5', '4',.....]. But if the entire list is the initial state None, this function will print a warning in the CLI console.

3.3.3 Build temperature sensor list

- List format: 2-dimension list, length [10][72], 10 rows for each room, 72 columns for each signal.
- Sampling interval: 1200s once, 72 times per room per day.
- Database query statement:

```
SELECT se.id_room, sd.value, sd.timestamp FROM sensor as se join stream_data as sd on se.id_sensor = sd.id_sensor
WHERE se.id_sensor_type = 4 and date(timestamp) =  + date_curr +  order by se.id_room, timestamp;
```

- Standardization of formats: We set the temperature to 2 ℃ as a step.

  For example：

  18.5 ℃ -> 18 ℃

  19.5 ℃ -> 18 ℃

  They are regarded as “no difference”, because they are in the same step.

3.3.4 Build power sensor list

- List format: 2-dimension list, 9 rows for each appliance, variable-length columns for each signal.
- Sampling interval:

There are 9 domestic appliances
1. Microonde : No.24, sampling interval is 30s
2. Televisione : No.26, sampling interval is 120s
3. HC2 Power : No.28, sampling interval is 300s
4. Frigorifero : No.32, sampling interval is 1200s, Power_level : 0w, 2w, 50w
5. Forno : No.34, sampling interval is 120s
6. Lavatrici : No.36, sampling interval is 120s
7. Serra A : No.45, sampling interval is 120s, directly record the original value
8. Lavastoviglie : No.148, sampling interval is 120s
9. PC : No.150, sampling interval is 120s, threshold: 5w

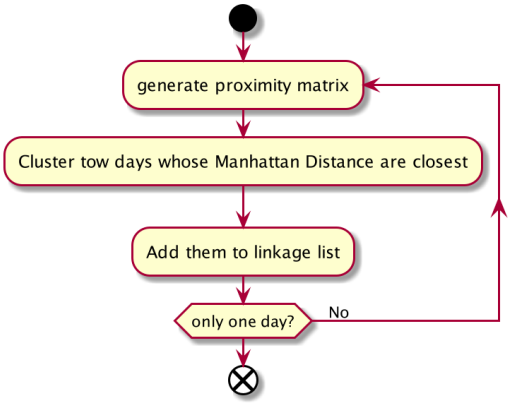
- Database query statement:

```
"SELECT se.id_sensor, sd.value, se.threshold, sd.timestamp FROM sensor as se join stream_data as sd on se.id_sensor = sd.id_sensor WHERE se.id_sensor_type = 5  and date(timestamp) =  + date_curr + order by se.id_sensor, timestamp;"
```

- Standardization of formats : For Microonde, Televisione, HC2 Power, Forno, Lavatrici, Lavastoviglie, PC, we record the “on & off” of these appliances according to the threshold, above the threshold is “on” (present with True), below the threshold is “off” (present with False). For Serra A, we just record the original value, and for Frigorifero, we record the power level. 0 ~ 2 is 0; 2 ~ 50 is 1, above 50 is 2.

3.4 Execute hierarchical clustering

This image is the process of the program to execute the hierarchical clustering algorithm. This process is a loop that continuously clusters the closest date while updating the proximity matrix.

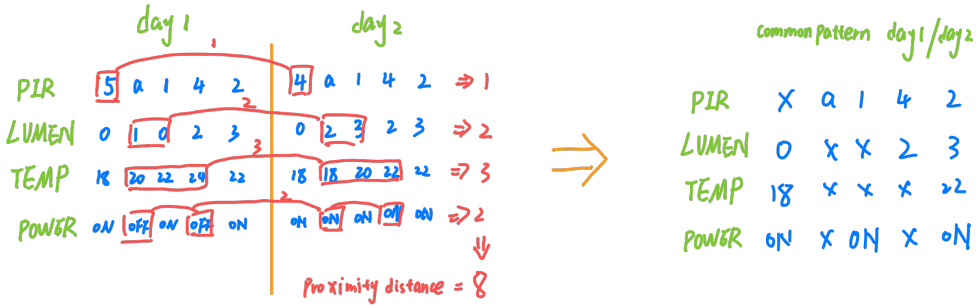


3.4.1 Defining Proximity between Clusters

We use Manhattan Distance to define proximity distance. For two lists, we iteratively compare their elements from beginning to end, if two elements are the same, the distance is unchanged. Opposite, if not equal, the distance is increased by one. Finally add up all distances of the available lists, which is the proximity distance of these two days. As shown in Figure A below.

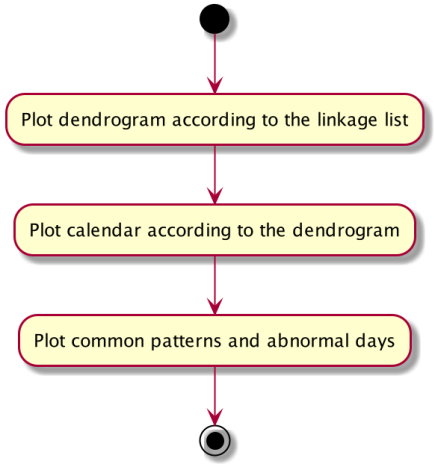
3.4.2 Cluster two days

If the two days are closest, they will be clustered, until clustered to only one day. So, in extreme cases, all elements are different, the maximum distance is equal to the length of all lists. In this program, it is 14040. The cluster process is shown in Figure B.



(A) (B)

3.5 Execute data visualization



Third-party libraries

- psycopg2 : To extract data from the database

  https://pypi.org/project/psycopg2/

- matplotlib.pyplot : To do data visualization

  https://matplotlib.org/index.html

- scipy.cluster.hierarchy: To plot dendrogram from linkage list

  https://docs.scipy.org/doc/scipy/reference/index.html