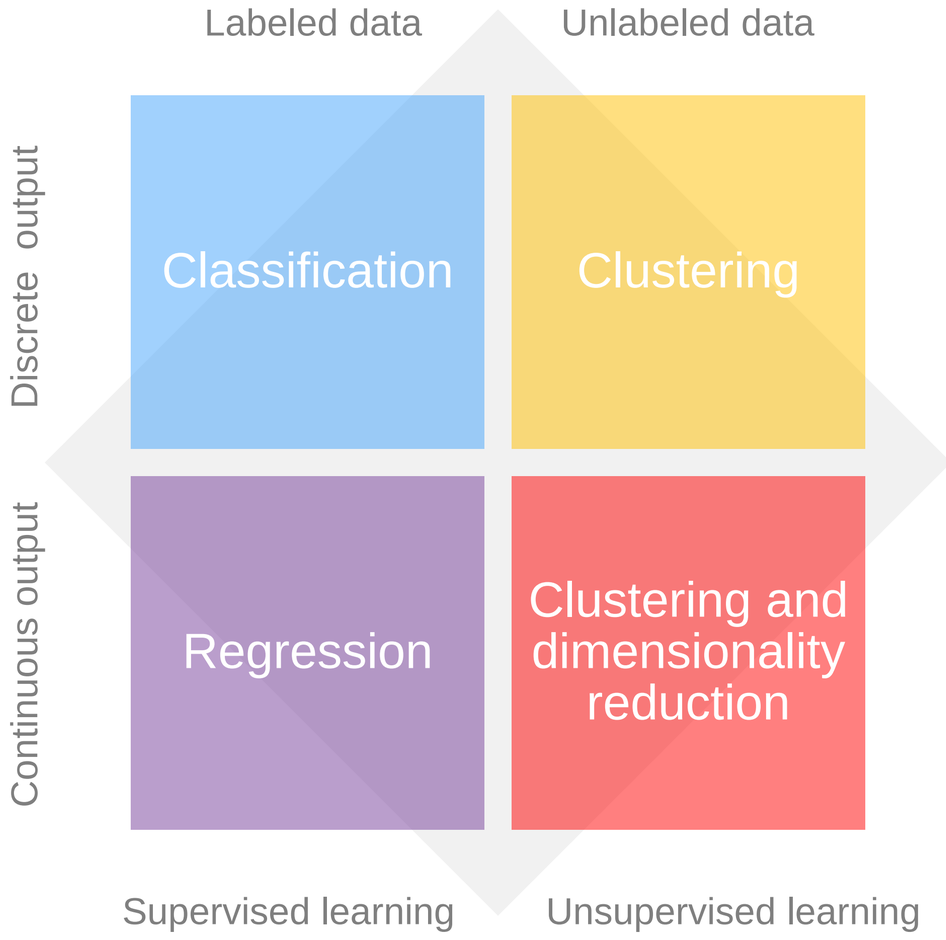
CREATIVE DATA MINING

Introduction to Python and Programming II

16.10.2017

Dr. Varun OJHA

Danielle GRIEGO



What we'll cover today

- Overview of exercise 1
- Revisit course schedule and objectives
- Introduce final project
- Recap from last week
- Python tutorial II

  - Functions, modules, reading and writing files

Exercise 01

There are 100 students in a degree program; each student may get a grade: bad, average, good, and excellent in mathematics, physics, and chemistry tests. Each student is given a total 180 minutes to finish all three tests. A teacher calculates each student intelligence quotient (IQ) based on the student's test results, the time taken by the student to finish the three tests, and the age of the students.

From the given paragraph, please identify the following:

1. What are the variables/features in the given problem? Distinguish discrete and continuous variables.

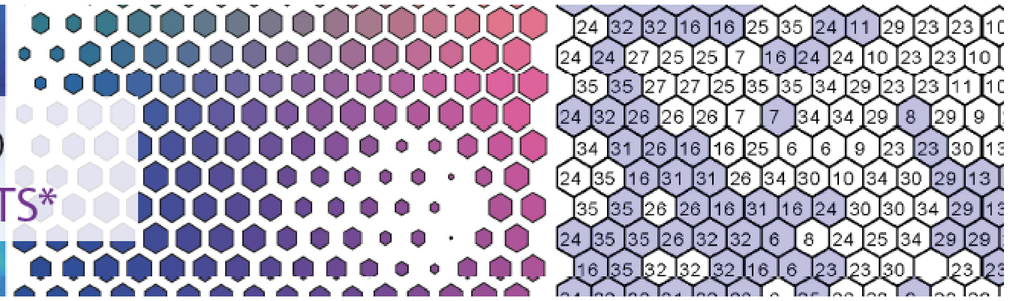
2. What are the input variables?

3. What is the target variable?

4. How many examples are in the given problem?

Mondays 10:00 – 12:00

052-0621-17L | 2 ECTS*



Creative Data Mining

The Creative data mining course aims to provide aspirants a hands-on experience on machine learning (ML) tools and techniques for data processing and analysis. Since future technologies increasingly rely upon the ML, urban systems and architecture shall adopt it and aspirant should learn creative ways to apply ML to better understand urban systems. The course covers a wider range ML techniques including supervised and unsupervised learning methods for data analysis and pattern recognition that help to better understand urban system for improving urban life.

All methods taught in the course will be applied to a common project to evaluate various dynamics of the urban environment. Students will work with time-series and geo-referenced data including temperature, relative humidity, illuminance, noise, people density, and dust particulate matter. Subjective impression survey data will also be integrated into the student projects to further explore influencing factors of the urban environment on our perceptual experiences. A selected neighborhood in the city of Zurich will be used as the case study and each student will present the findings of their research question in a final project.

Additionally, there are two of non-architectural skills the participants can develop during this course. First is an introduction to programming where at a minimum they can successfully copy and paste code-snippets to customize the computational tools presented in the course. Second, how clustering methods like PCA or K-Means could be applied in an architectural context.

Where

HIT H 31.4 (Video wall)

25.09.2017	Introduction to the knowledge discovery process
02.10.2017	Fundamentals of supervised machine learning
9.10.2017	Introduction to python & programming I
16.10.2017	Introduction to python & programming II Introduce final projects
23.10.2017	Seminar Week- No Lecture
30.10.2016	Supervised learning problem solving in python (MLP & SVM)
06.11.2016	Fundamentals of unsupervised learning (K-means, DBSCAN, PCA)
13.11.2017	Unsupervised learning problem solving in python
20.11.2017	Review data and examples for final projects
27.11.2017	Project proposal discussions
04.12.2017	Exploration of Real-World problems Q&A Workshop I
11.12.2017	Q&A Workshop II
18.12.2017	Final Critique

Course objectives and outcomes

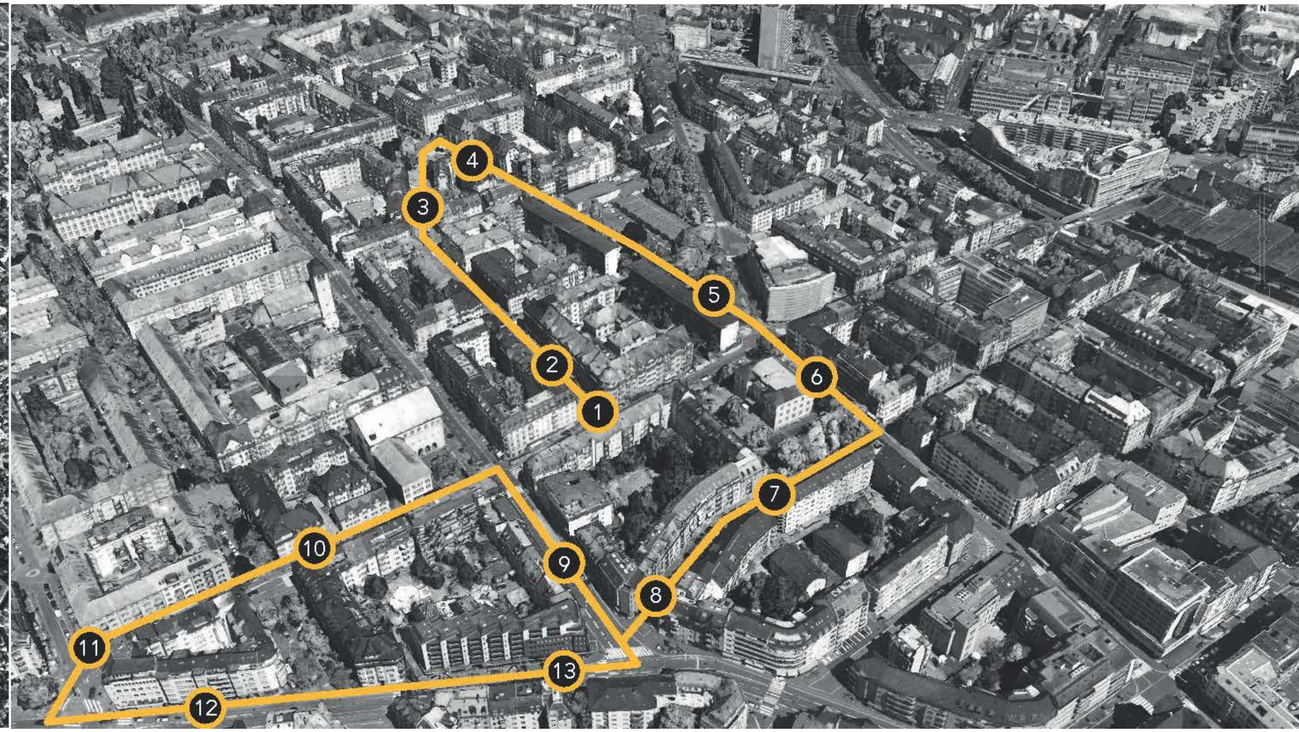
1. Become familiar with programming and integrating new tools in your work
2. Develop an interesting “research” question and learn how to answer it by:

- Selecting appropriate data
- Applying relevant analysis and visualization techniques
- Interpreting and refining your results

Final projects: Possible dataset

ESUM- Analyzing trade-offs between Energy and Social performance of Urban Morphologies

Location Wiedikon Zürich



14 survey checkpoints along experimental path

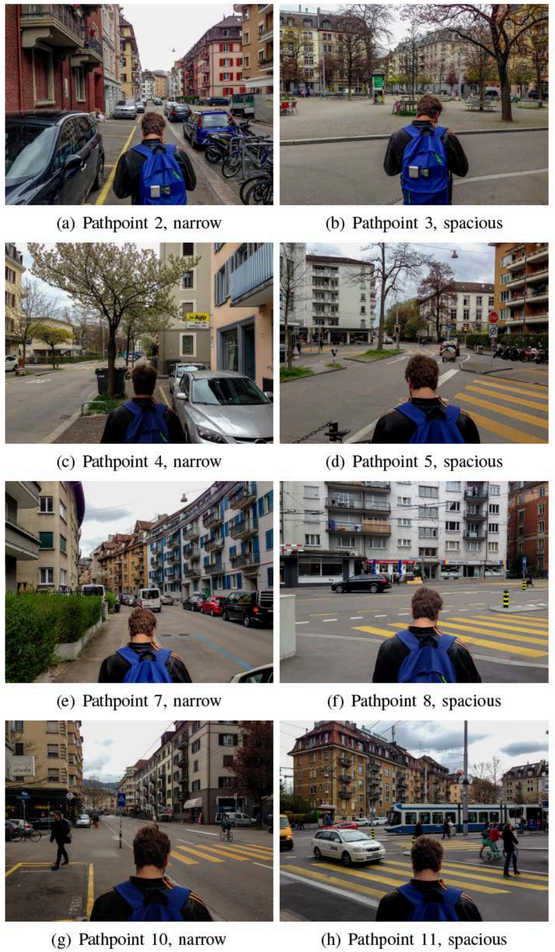
FNSNF
Swiss National Science Foundation

Example

ESUM- Analyzing trade-offs between Energy and Social performance of Urban Morphologies

(Raw) data from 37 participants:

- Investigate impact of static (urban morphology) and dynamic features (environmental sensors) of the built environment on perception (using surveys and biofeedback data)



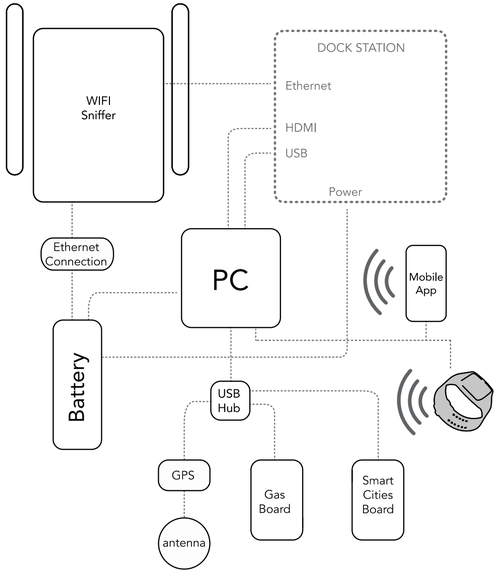
(a) Pathpoint 2, narrow (b) Pathpoint 3, spacious

(c) Pathpoint 4, narrow (d) Pathpoint 5, spacious

(e) Pathpoint 7, narrow (f) Pathpoint 8, spacious

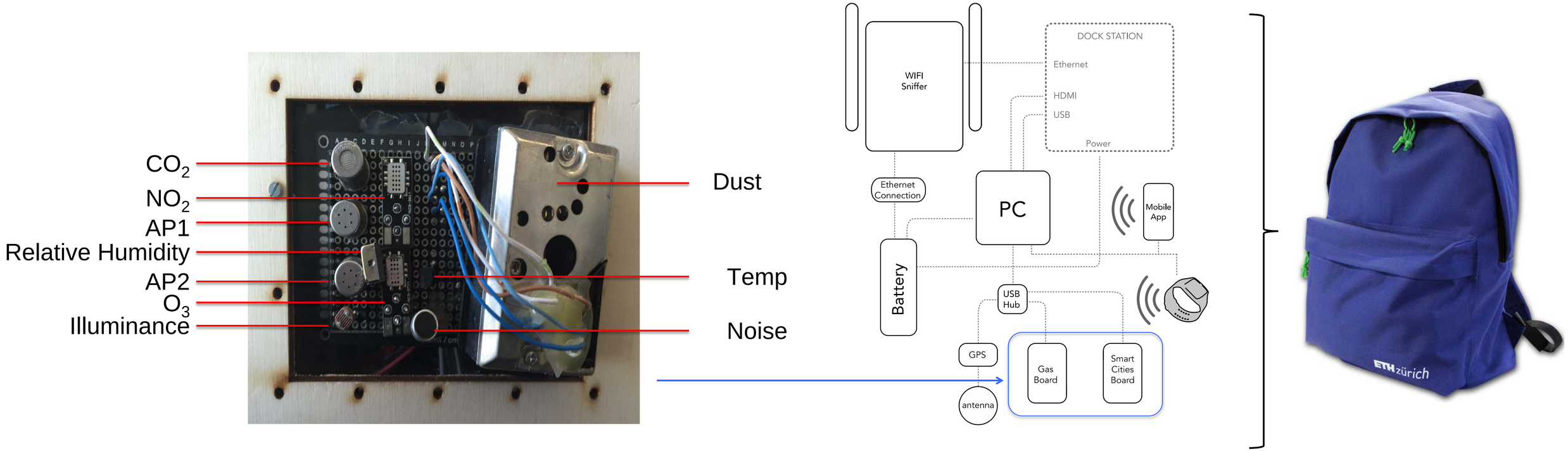
(g) Pathpoint 10, narrow (h) Pathpoint 11, spacious

Fig. 2. Four instances of narrow-spacious spatial configurations and their corresponding pathpoints along the select path.



Mobile sensor equipment

Sensorbackpack with environmental and position sensors



Mobile Sensor equipment

Biofeedback wristband

E4 Sensors

PPG Sensor
Photoplethysmography Sensor - Measures Blood Volume Pulse (BVP), from which heart rate, heart rate variability (HRV), and other cardiovascular features may be derived

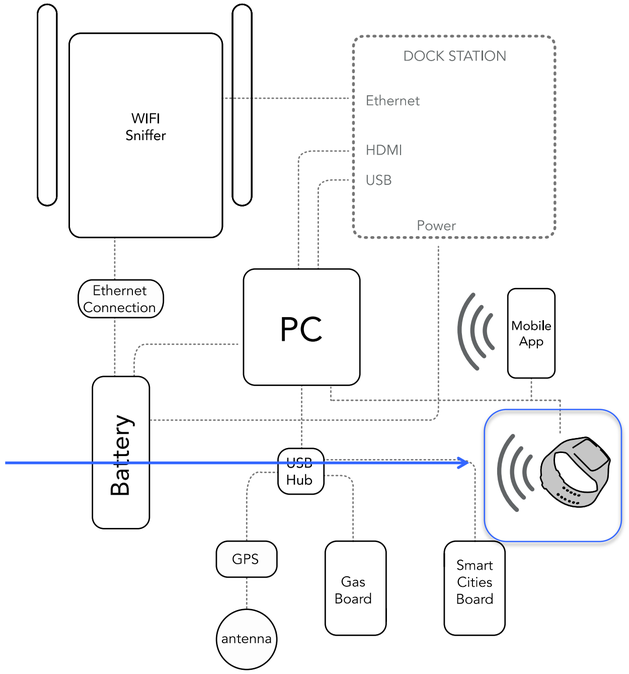
3-axis Accelerometer
Captures motion-based activity

Event Mark Button
Tags events and correlate them with physiological signals

EDA Sensor (GSR Sensor)
Electrodermal Activity Sensor - Used to measure sympathetic nervous system arousal and to derive features related to stress, engagement, and excitement.

Infrared Thermopile
Reads peripheral skin temperature

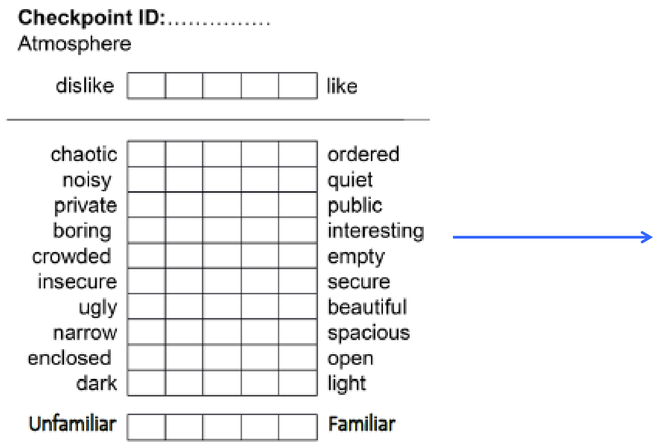
Internal Real-Time Clock
Temporal resolution up to 0.2 seconds in streaming mode

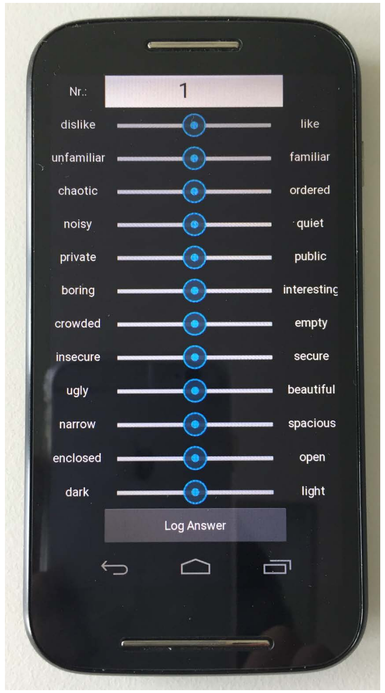


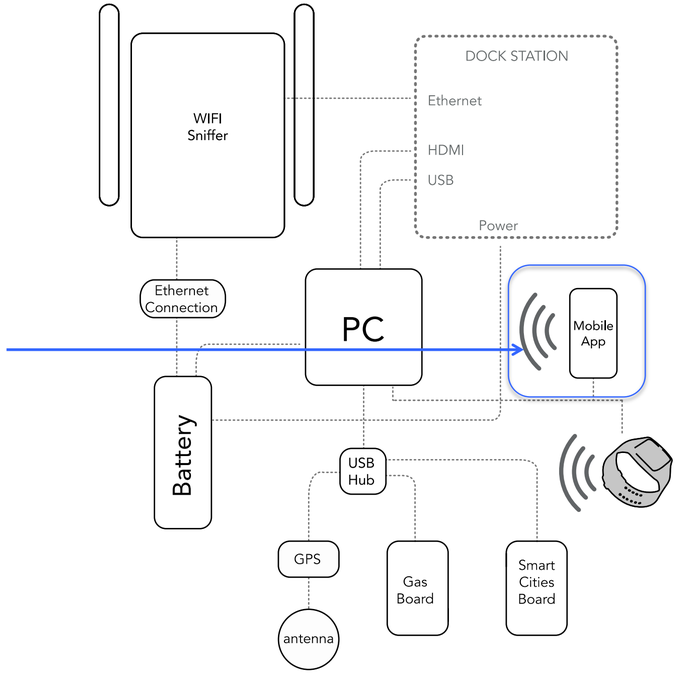
https://www.empatica.com/e4-wristband

Mobile Sensor equipment

Biofeedback wristband







Experimental data-set

Environmental features

Noise (soundscape) sensor: Smart Cities Board
Manufacturer: Libelium Comunicaciones Distribuidas S.L., Zaragoza (Spain)
Measurement range:50-100 dB
Frequency: 0.4 Hz
Accuracy: ± 2.5 dB
Physical meaning: Visit

Dust sensor: Smart Cities Board
Manufacturer Libelium Comunicaciones Distribuidas S.L., Zaragoza (Spain)
Measurement range: 0.5V/(0.1 mg/m3)
Frequency: 0.4 Hz
Accuracy: Operating supply voltage 5±0.5V

Temperature sensor: HOBO U12 Logger
Manufacturer: Onset Computer Corporation, Bourne, MA, USA
Measurement range: -20°C -70°C
Frequency: 1 Hz
Accuracy: ± 0.35°C from 0°C to 50°C

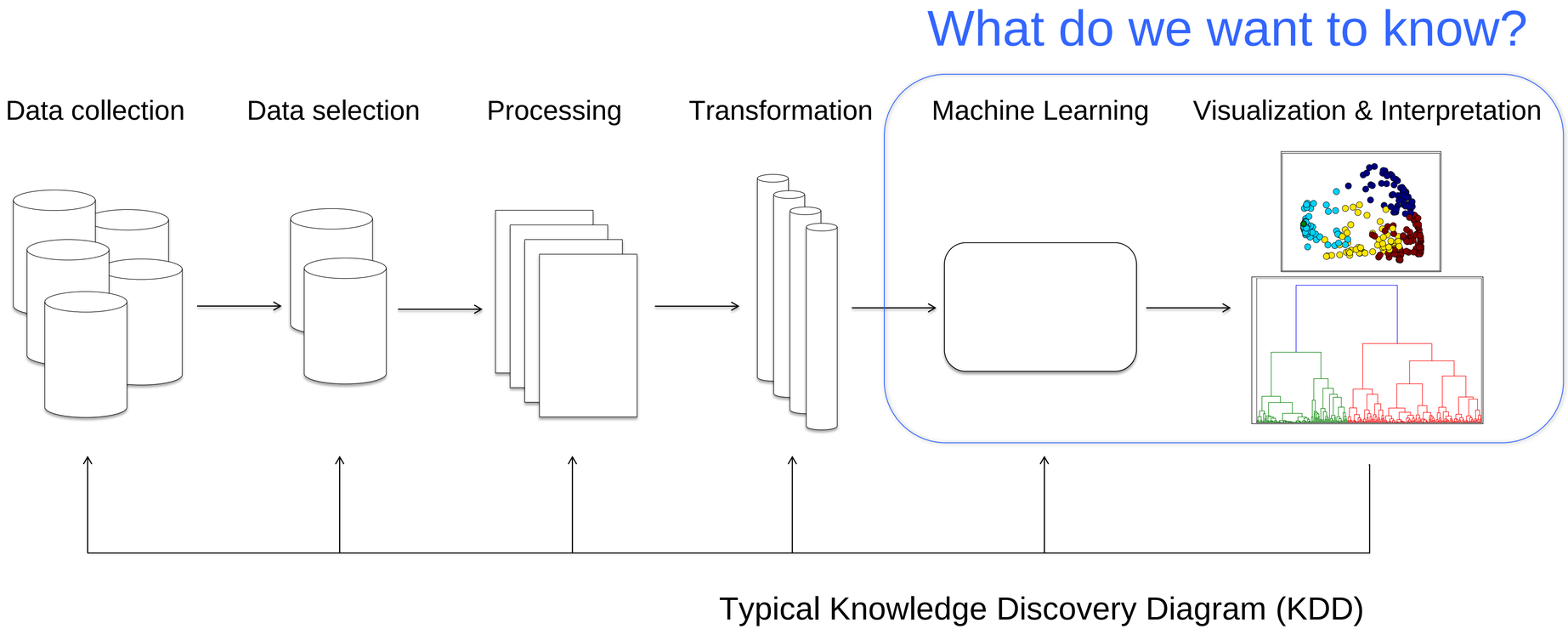
Relative Humidity Sensor: HOBO U12 Logger
Manufacturer: Onset Computer Corporation, Bourne, MA, USA
Measurement range: 5%-95% (non-condensing)
Measurement unit: %
Frequency: 1 Hz
Accuracy: : ± 2.5% typical, ± 3.5% max

Illuminance sensor: HOBO U12 Logger
Manufacturer: Onset Computer Corporation, Bourne, MA, USA
Measurement range: 10-32,200 lux
Frequency: 1 Hz
Physical meaning: Visit

Global Navigation Satellite System (GNSS)/ Global Positioning System (GPS): Ducat 10 uBlox NEO-M8N, TW2410 Antenna
Manufacturer: Tallysman Wireless Inc.
Measurement: WGS84 spherical coordinates
Frequency: 1Hz
Accuracy: Visit

Background

How can data mining be creative?

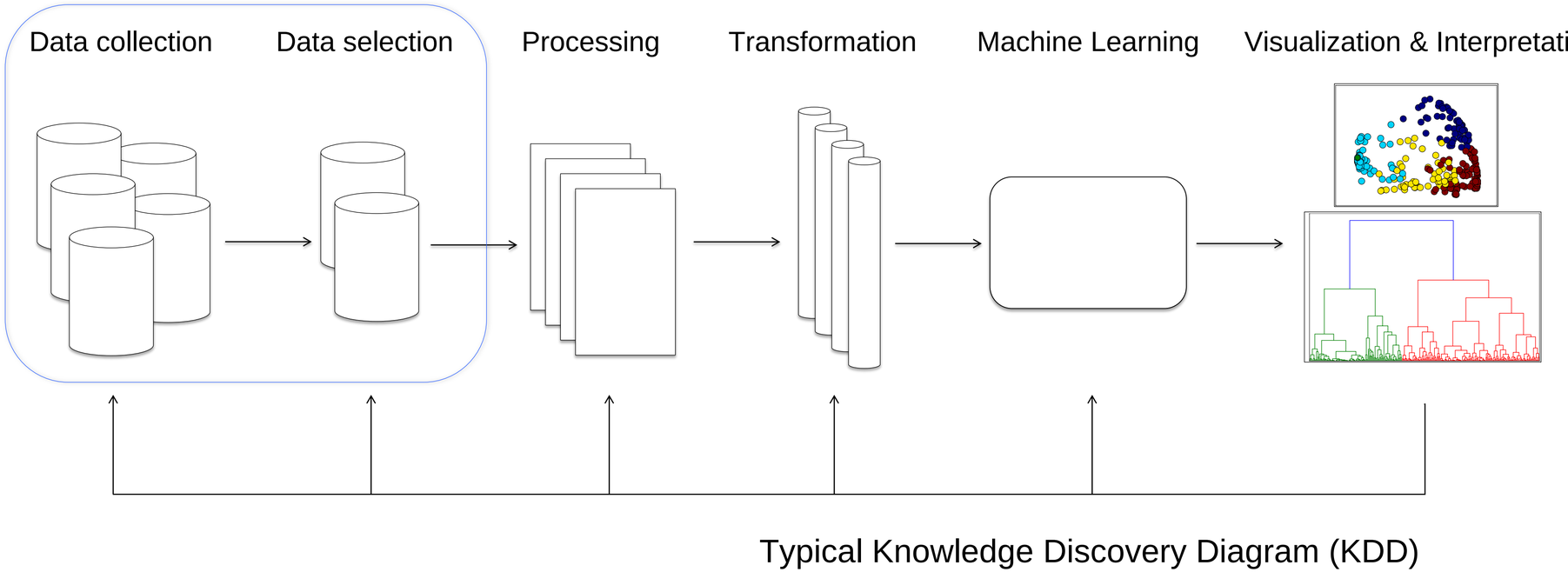


Typical Knowledge Discovery Diagram (KDD)

Background

How can data mining be creative?

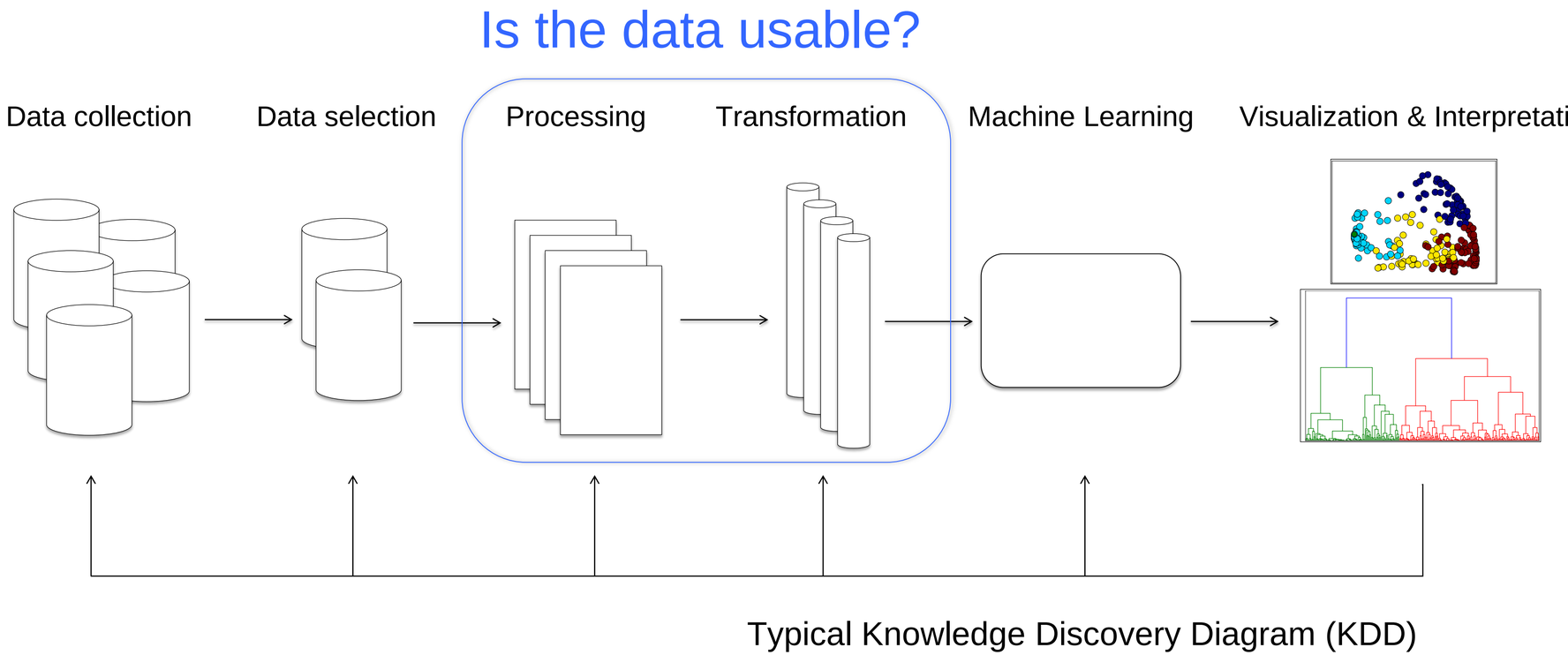
Domain specific data source(s)



Typical Knowledge Discovery Diagram (KDD)

Background

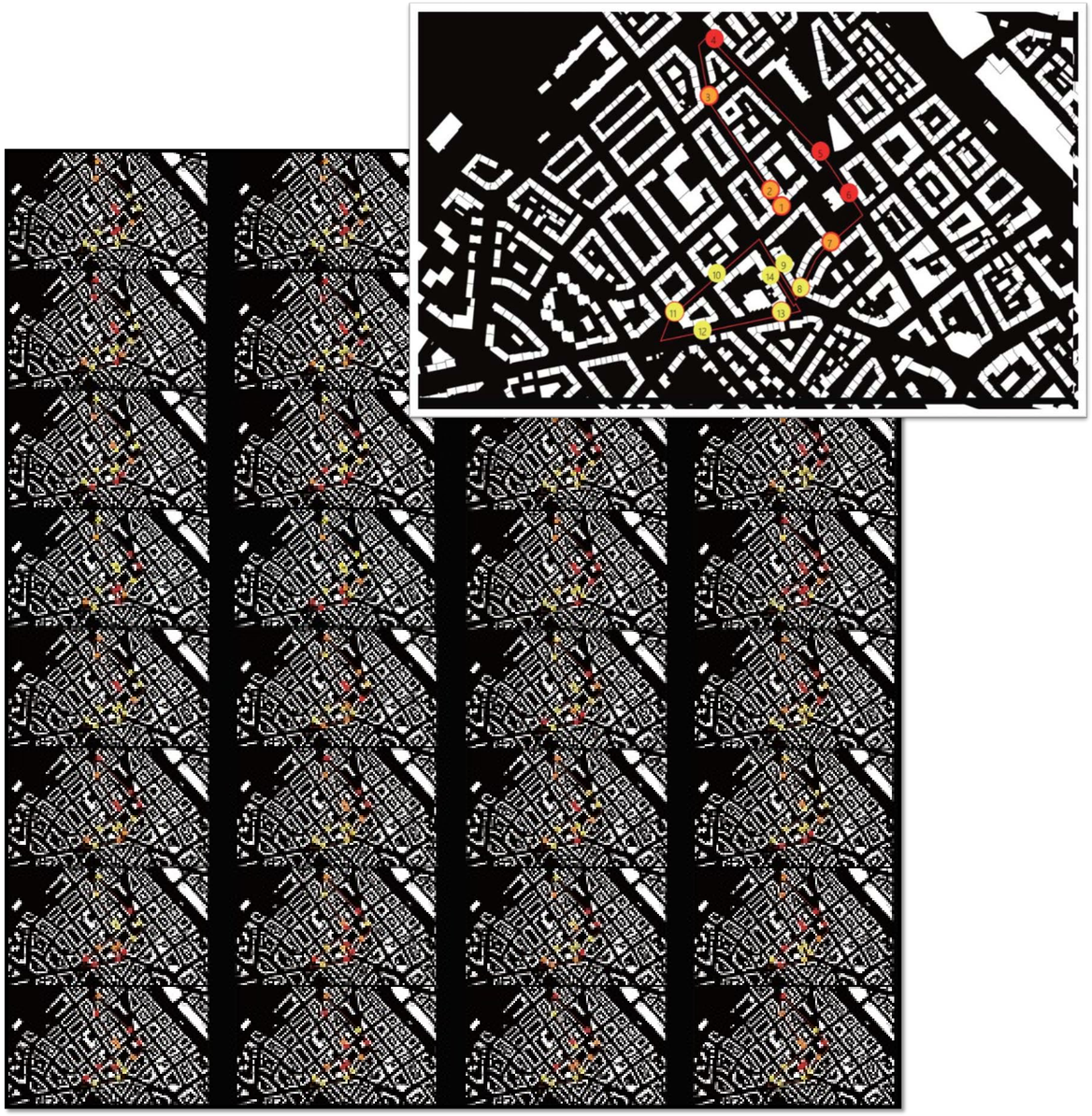
The not-so creative, but essential part of data mining



Typical Knowledge Discovery Diagram (KDD)

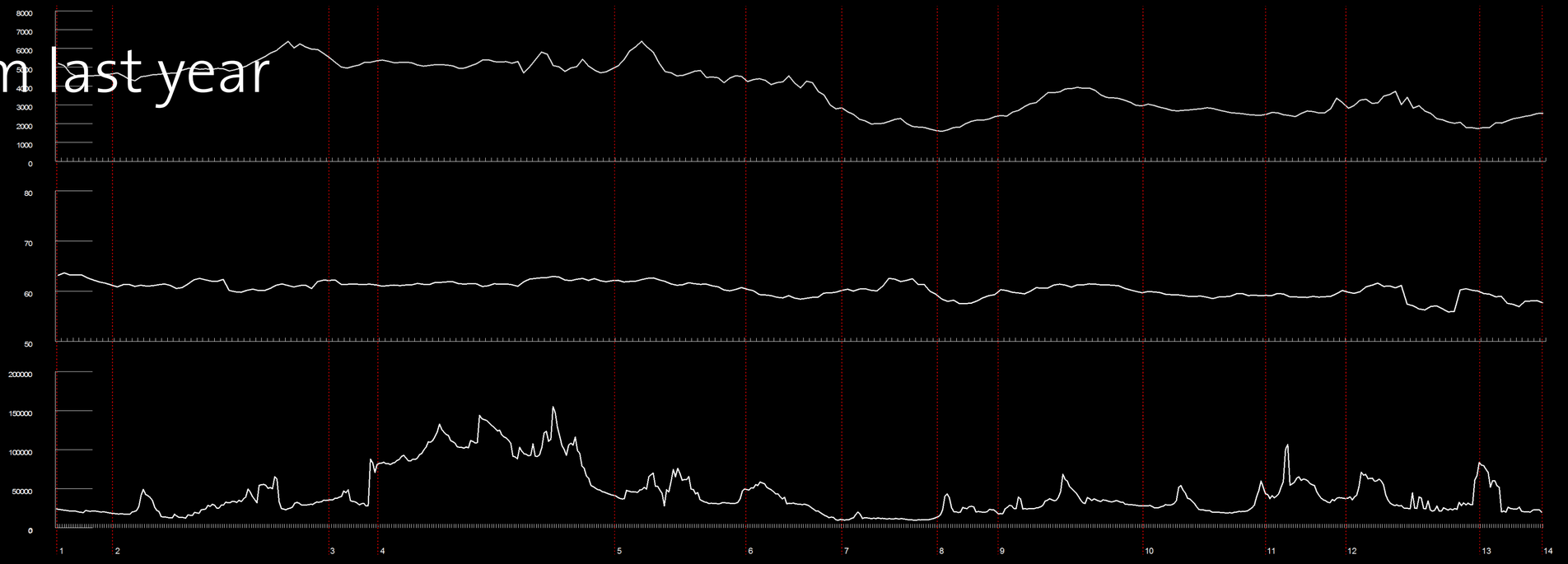
Final Project Description

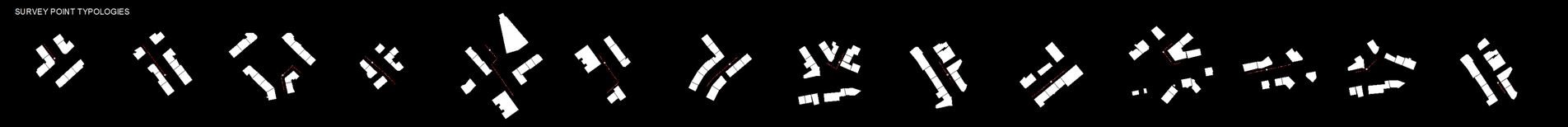
1. Formulate 1-2 specific question(s) of interest to you
2. State your hypothesis/expected outcome based on supporting literature (minimum one source) your expertise, and intuition
3. Answer that question through your analysis, for this:
   - Select the best available data sources for your question (min. of 2 data sources)
   - Include at least one supervised learning or unsupervised learning technique
4. Summarize your results
   - Does your analysis answer your question(s)?
5. Conclusions & lessons learned
   - Fore example if you had more time, data, resources, etc. how would you improve your study
6. Include motivation and references

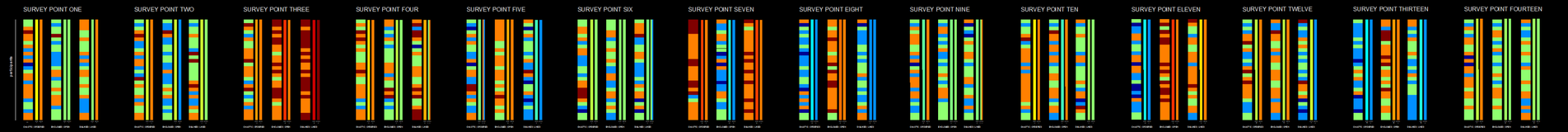


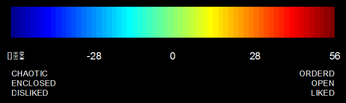
Images: Final project CDM Spring 2017 by Biyu Wang & Jiani Liu

One example from last year









DARCH

iA Chair of Information Architecture

Final project from Jochen Aarts and Stéphane de Weck Creative Data Mining FS2016

What is programming?

- Programming is about solving problems and puzzles. You describe a precise set of instructions which the computer follows exactly.

For example if we want to sort a list:

```
list=[23, 44, 5, 17, 8, 90, 102]
list.sort()
print(list)

output: [5, 8, 17, 23, 44, 90, 102]
```

Types and Values

- *Values* are for example: 5, 12.6, True or “Hello world”
- Each value is of a certain *type*
  - Numbers are: →numeric (integer or float)
  - True is: →boolean (only two values or states)
  - “Hello world” is: →a string (list of characters)

Operators

- Operators represent a value manipulation (+, -, /, %, ….)
- It is important to check that the data type is correct, otherwise you might get an unexpected result.

```
p=int(17/9)
print(p)

p=round(17/9)
print(p)

p=float(17/9)
print(p)
```

```
Output:

1

2

1.8888888888
```

Data Structures

There are three basic data structures in Python:

1. Lists- A sequence of values of any type

Example:

```
CDM=['There are', [20, 17, 11, 7],'students registered and', 2, 'instructors']
```

2. Lists provide several functionalities:

```
CDM[2]          #Accesses the third value
CDM.remove(2)   #Removes the first occurrence of 2
CDM.append(2)   #Adds 2 to the end of the list
```

Data Structures

There are three basic data structures in Python:

2. Tuples- a sequence of values of any type, however cannot be altered after it is initiated.

Example:

```
CDM=('There are', [20, 17, 11, 7],'students registered and', 2,
'instructors')
```

Vs.

```
CDM=['There are', [20, 17, 11, 7],'students registered and', 2,
'instructors’]
```

Data Structures

There are three basic data structures in Python:

3. Dictionaries- store data as key-value pairs.
   Each key has an associated value.

Example

```
CDM={'Number of students so far': [20, 17, 11, 7]}

CDM['Number of students so far']
# Returns [20,17,11,7]

CDM['final number']='usually 1/3 of first day’
#Adds a new element with key ‘final number’ and value ‘usually 1/3 of
the first day’
```

Logical Operators

Logical Operators are expressions that evaluate to either *True* or *False*.

They are normally used in conditional statements.

Relational operators: ==, !=, <, >, <=, >=, and, or, not

They stand for: equal, not equal, smaller, greater, smaller equal, greater equal, and, or, not

Examples:

x < 2 or x >= 0

x == 0 or x != 0

not (x == 0 or x > 100)

Functions

If you do not want to retype the same code over and over again, you can define a function that you can call over and over again

Structure:

def funtionName(parameters):

    instructions

    return variable

Example:

```
def square(a):
    a = a * a
    return a

# To get the square of 6, now just write:

square(6)
```

Built-in python Functions

The Python interpreter has a number of functions and types built into it that are always available. They are listed here in alphabetical order.

		Built-in Functions		
abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	
delattr()	hash()	memoryview()	set()	

https://docs.python.org/3/library/functions.html

Modules

- Often, more complicated code is already implemented and can be used as modules. They help to be more efficient, because the programmer does not need to implement everything from scratch.
- When installed, modules can be integrated in the code using the keyword *import*.

Example:

```
import pandas as pd
# imports the Pandas module, and now you can access the built in functions
```

Working with Files

Reading in a file using pandas:

```
 d=pd.read_csv('filename.csv', header=None)
```

Creating dataframes:

```
df=pd.Dataframe(data=d)
```

23:30.4	1	1	-2	1	-2	-1	-1	0	2	-1	2	1	1
24:57.0	2	1	-2	1	-1	-1	1	1	2	0	1	1	1
29:03.1	3	2	-2	-1	2	1	1	-1	2	1	2	2	1
30:37.1	4	2	-1	1	0	1	0	2	2	1	2	2	1
34:24.9	5	-1	-1	-1	-2	2	1	-1	2	-1	1	2	0
36:37.1	6	1	0	2	-1	1	-1	1	2	1	1	1	1
38:38.9	7	1	-1	-1	-1	1	1	1	2	1	0	1	1
40:37.8	8	-1	1	-1	-2	1	-1	-1	2	-1	1	1	2
41:44.5	9	1	-1	1	-1	1	1	-1	2	1	-1	1	1
44:48.4	10	1	-1	1	0	1	-1	-1	2	-1	1	1	1
46:38.4	11	1	1	-2	-1	1	-1	-1	2	-1	1	1	0
48:04.0	12	1	-1	1	-1	2	1	-1	2	0	2	1	0
50:28.5	13	-1	1	-2	-2	2	0	-2	2	-1	1	1	1
51:56.2	14	1	2	1	-1	2	-1	1	2	2	2	1	1

Appending column names:

```
columnNames=['Date-Time', 'path-point', 'like-dislike', 'familiar-unfailiar', 'ordered-chaotic', 'quiet-noisy', 'public-private',
'interesting-boring', 'empty-crowded', 'secure-insecure', 'beautiful-ugly', 'spacious-narrow', 'open-enclosed', 'light-dark']

df.columns = columnNames
```

Manipulating dataframes

For Example, from the previous dataset apply the Transpose Attribute:

```
transpose= df.T[1:14]
print(transpose)
```

Writing files into a folder:

```
import os
#hint the os module will also help with your hw

os.makedirs(r'./Single-survey/')  #creates the directory
newpath = r'./Single-survey/‘      #creates access

Transpose.to_csv(newpath +‘transpose.csv‘)
#now look in your working directory to find the new csv file
```

Attributes

`T`	Transpose index and columns
`at`	Fast label-based scalar accessor
`axes`	Return a list with the row axis labels and column axis labels as the only members.
`blocks`	Internal property, property synonym for as_blocks()
`dtypes`	Return the dtypes in this object.
`empty`	True if NDFrame is entirely empty [no items], meaning any of the axes are of length 0.
`ftypes`	Return the ftypes (indication of sparse/dense and dtype) in this object.
`iat`	Fast integer location scalar accessor.
`iloc`	Purely integer-location based indexing for selection by position.
`is_copy`	
`ix`	A primarily label-location based indexer, with integer position fallback.
`loc`	Purely label-location based indexer for selection by label.
`ndim`	Number of axes / array dimensions
`shape`	Return a tuple representing the dimensionality of the DataFrame.
`size`	number of elements in the NDFrame
`style`	Property returning a Styler object containing methods for building a styled HTML representation fo the DataFrame.
`values`	Numpy representation of NDFrame

Methods

`abs`()	Return an object with absolute value taken–only applicable to objects that are all numeric.
`add`(other[, axis, level, fill_value])	Addition of dataframe and other, element-wise (binary operator *add*).
`add_prefix`(prefix)	Concatenate prefix string with panel items names.
`add_suffix`(suffix)	Concatenate suffix string with panel items names.
`agg`(func[, axis])	Aggregate using callable, string, dict, or list of string/callables
`aggregate`(func[, axis])	Aggregate using callable, string, dict, or list of string/callables

https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.html

Programming assignment

Data Processing: data transformation

Transform the survey data from each ESUM participant by survey question:

- The output from each participant's survey responses is a single dataframe (csv file); each participant answered 12 questions at each of the 14 check points
- Your task is to create a program which automatically generates a single dataframe (csv file) for each of the 12 questions; you should have 12 csv files (32 participants by 14 check points with the populated survey results)
- Write the files to a single folder and include this with the python file in a zip folder and email it to us by 9am October 30th

Conceptual assignment

Research other examples of urban data mining and make 2 slides about the most interesting project/application/research group(s) that you find. This will be presented at the beginning of next lecture

- Please send your slides to us by 9am on October 30th

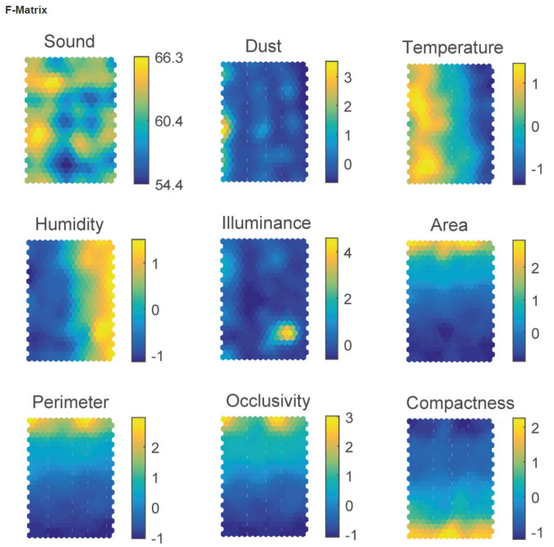


Fig Each feature was linearly scaled with a variance of one so that they have equal importance in computing distance and influence in clustering on the map.

Thank you!

Varun Ojha ojha@arch.ethz.ch

Danielle Griego griego@arch.ethz.ch