Time Series

4 Smart City Applications

Enrico Collini

Corso di: Big Data Architectures

Prof. Paolo Nesi
Dep DINFO, University of Florence
Via S. Marta 3, 50139, Firenze, Italy
DISIT Lab, Sistemi Distribuiti e Tecnologie Internet
http://www.disit.dinfo.unifi.it/ , Https://www.disit.org
paolo.nesi@unifi.it http://www.disit.dinfo.unifi.it/nesi

Join at menti.com | use code 1849 9427

Mentimeter

Instructions

Go to

www.menti.com

Enter the code

1849 9427

Or use QR code

Time Series
4 Smart City Applications

Enrico Collini

Lecture Schedule (with me)

- 27/11 (Today!)
  - Time Series Basics & the OSEMN pipeline
  - Road To Time Series Forecasting
  - Deepening into Time Series Preparation for AI
    - Short-Term Prediction of Bikes Availability on Bike-Sharing Stations
    - Long Term Predictions of Yearly NO_2 in the Florence Metropolitan Area
    - Short-term prediction of city traffic flow via convolution
  - Time Series Data Analysis Workshop
  - Interpret Data Analytics using Snap4City
    - » Retrieve Data from Snap4City Open APIs
    - » Data Analytic Node-Red Python Container
    - » Data Analytics MyKpi + Monitoring Dashboard

UNIVERSITÀ DEGLI STUDI FIRENZE | DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE | DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

SNAP4CITY

```
collini@ubuntu-Precis-------------------ck:~$ who
collini   pts/16       202--------------------------------------------------)
```

almost 3 years

Enrico Collini

AI Smart City solutions Research Fellow at Distributed System and Internet Technologies Lab (DISIT), Università Degli Studi di Firenze

UNIVERSITÀ DEGLI STUDI FIRENZE | DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE | DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB DISTRIBUTED DATA INTELLIGENCE AND TECHNOLOGIES LAB

start 3rd year

UNIVERSITÀ DI PISA

DOTTORATO NAZIONALE IN INTELLIGENZA ARTIFICIALE

Time Series Lecture

- What is time series Data?
- What can you do with time series analysis (TSA)
- Stepladder to conduct a great time series analysis… with examples

Time Series Data

A collection of observations obtained through repeated measurements of time

- Each instant represents a **timestep**
- The values associated with that time are the **attributes**
- The data typically arrives in time order
- Time-intervals can be **regular** (metrics) or irregular (events)

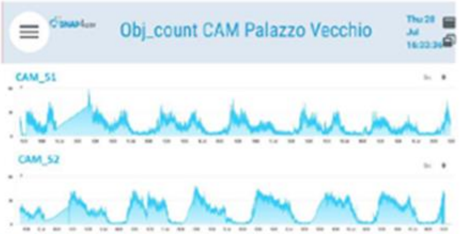
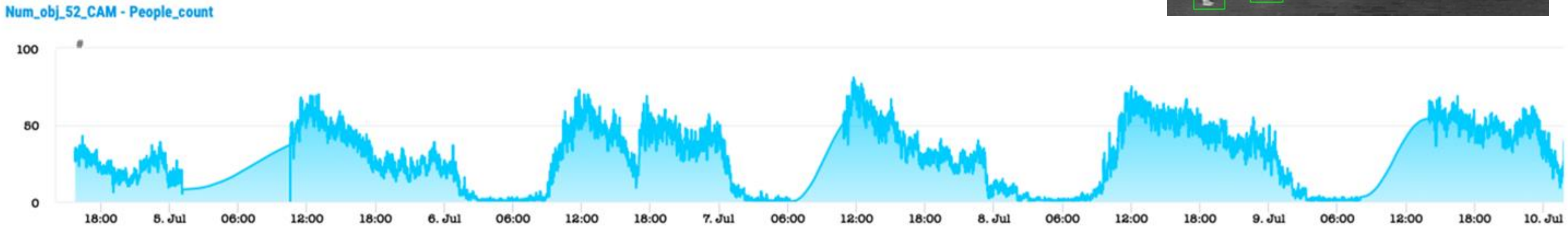


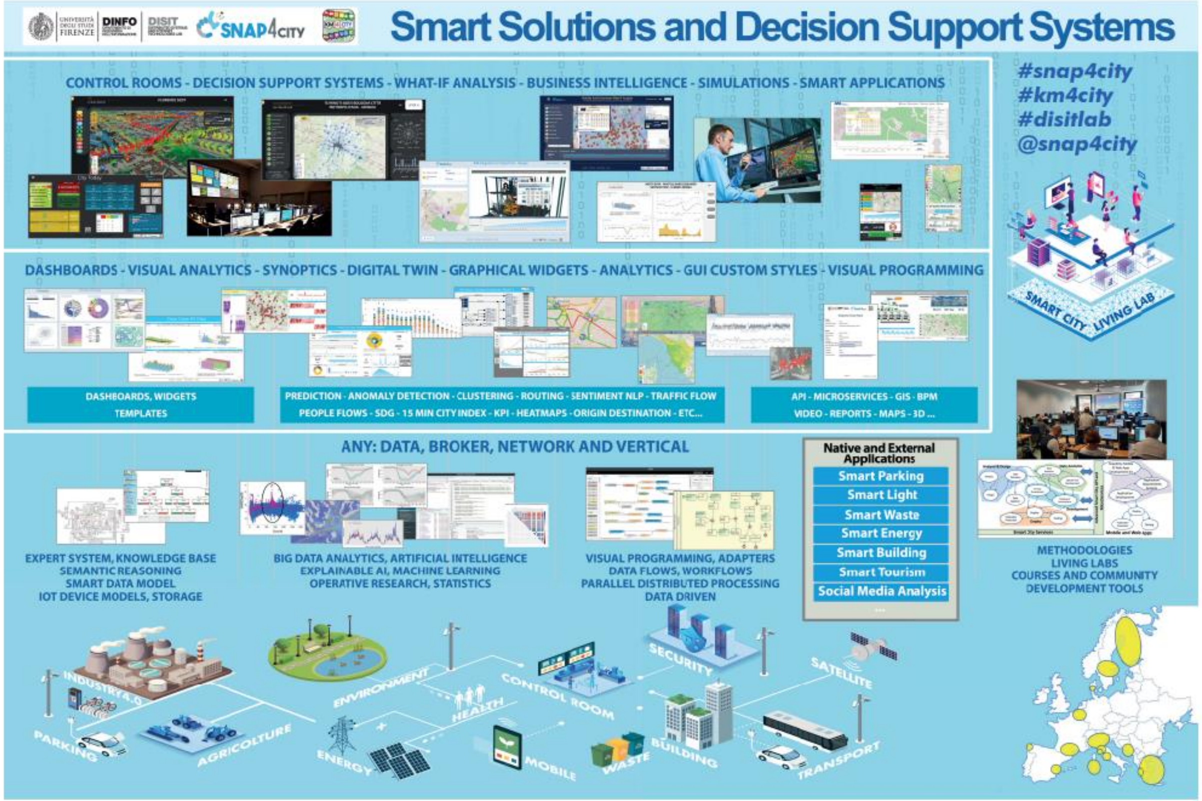
FIGURE 11. Monitoring Dashboard of people counting in Piazza Della Signoria, Florence



Introduced the definition of Time-Series Data what example comes into your mind of this type of data?

Waiting for responses ...


BEEP
BEEP


Smart Solutions and Decision Support Systems
#snap4city
#km4city
#disitlab
@snap4city
CONTROL ROOMS - DECISION SUPPORT SYSTEMS - WHAT-IF ANALYSIS - BUSINESS INTELLIGENCE - SIMULATIONS - SMART APPLICATIONS
DASHBOARDS - VISUAL ANALYTICS - SYNOPTICS - DIGITAL TWIN - GRAPHICAL WIDGETS - ANALYTICS - GUI CUSTOM STYLES - VISUAL PROGRAMMING
SMART CITY LIVING LAB
DASHBOARDS, WIDGETS
TEMPLATES
PREDICTION - ANOMALY DETECTION - CLUSTERING - ROUTING - SENTIMENT NLP - TRAFFIC FLOW
PEOPLE FLOWS - SDG - 15 MIN CITY INDEX - KPI - HEATMAPS - ORIGIN DESTINATION - ETC...
API - MICROSERVICES - GIS - BPM
VIDEO - REPORTS - MAPS - 3D ...
ANY: DATA, BROKER, NETWORK AND VERTICAL
Native and External Applications
Smart Parking
Smart Light
Smart Waste
Smart Energy
Smart Building
Smart Tourism
Social Media Analysis
...
EXPERT SYSTEM, KNOWLEDGE BASE
SEMANTIC REASONING
SMART DATA MODEL
IOT DEVICE MODELS, STORAGE
BIG DATA ANALYTICS, ARTIFICIAL INTELLIGENCE
EXPLAINABLE AI, MACHINE LEARNING
OPERATIVE RESEARCH, STATISTICS
VISUAL PROGRAMMING, ADAPTERS
DATA FLOWS, WORKFLOWS
PARALLEL DISTRIBUTED PROCESSING
DATA DRIVEN
METHODOLOGIES
LIVING LABS
COURSES AND COMMUNITY
DEVELOPMENT TOOLS
INDUSTRY 4.0
ENVIRONMENT
HEALTH
CONTROL ROOM
SECURITY
SATELLITE
PARKING
AGRICOLTURE
ENERGY
MOBILE
WASTE
BUILDING
TRANSPORT


POLLUTION
ECOLOGY

Time Series Data Analysis (TSA)

What can you do with time-series data?

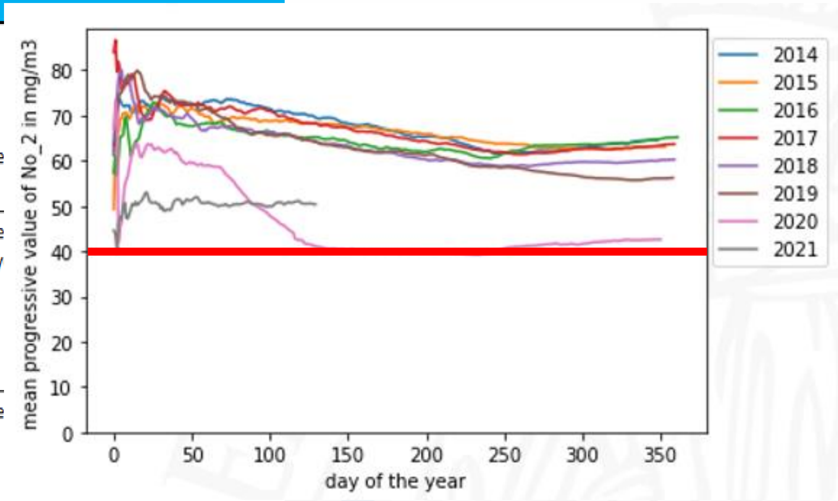
Analyze change (past - present -future)

3 main analysis types:

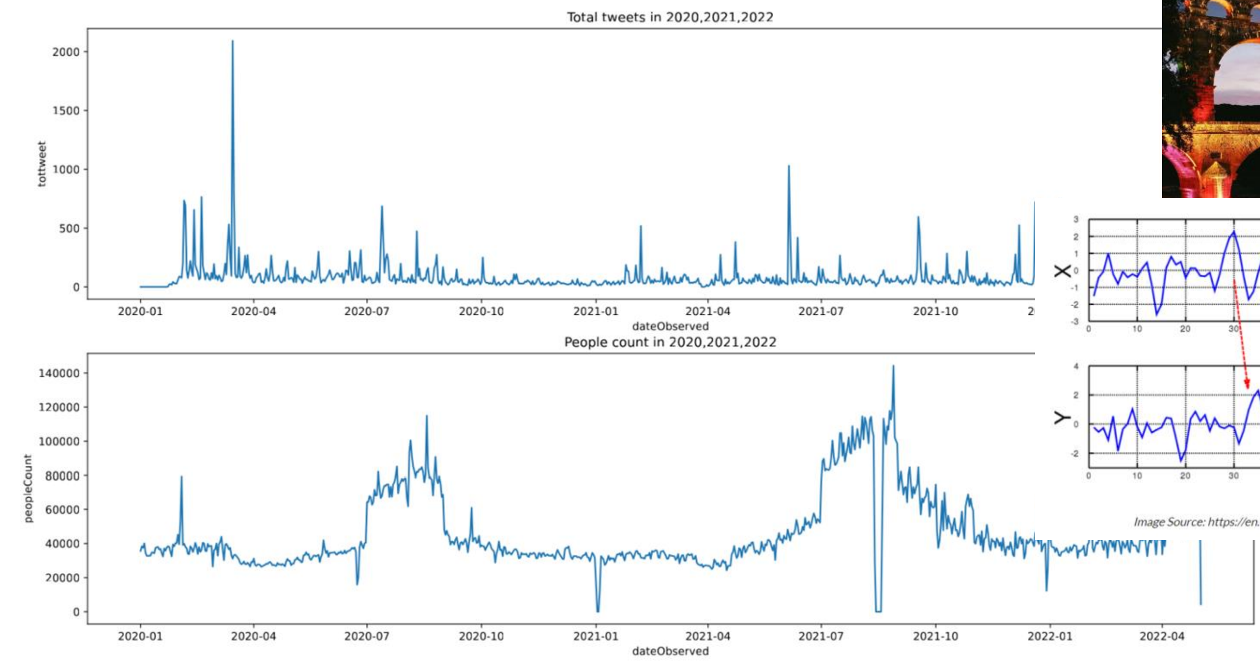
A) Access the impact of a single event (descriptive)
B) Study the interaction between a set of values
C) Forecast Future Values of a Time-Series using the previous values of one series (or also values from others) (prediction)

A) Access the impact of a single event

	EU Air Quality Directives			
Pollutant	**Averaging period**	**Objective**	**Concentration**	**Comments**
$PM_{2.5}$	24-hour	Target value		
$PM_{2.5}$	Annual	Limit value	25 µg/m³	
$PM_{2.5}$	Annual	Indicative limit value	20 µg/m³	
PM_{10}	24-hour	Limit value	50 µg/m³	Not to be exceede
PM_{10}	Annual	Limit value	40 µg/m³	
O_3	Max. daily 8-hour mean	Target value	120 µg/m³	Not to be exceede (averaged over 3 y
O_3	Max. daily 8-hour mean	Long-term objective	120 µg/m3	
O_3	8-hour	Target value		
O_3	Peak season [a]	Target value		
NO_2	Hourly	Limit value	200 µg/m³	Not to be exceede
NO_2	Annual	Limit value	40 µg/m³	
NO_2	24-hour	Target value		
SO_2	Hourly	Limit value	350 µg/m³	Not to be exceeded on more than 24 hours/year
SO_2	24-hour	Limit value	125 µg/m³	Not to be exceeded on more than 3 days/year
CO	Max. daily 8-hour mean	Limit value	10 mg/m³	
CO	24-hour	Target value		



B) Study the interaction between a set of values



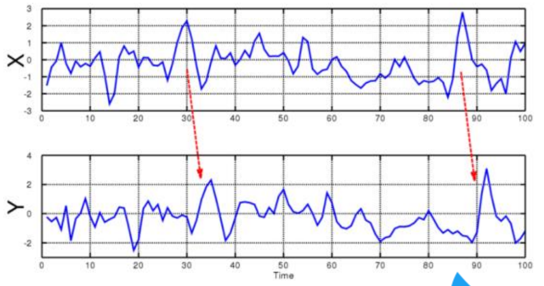
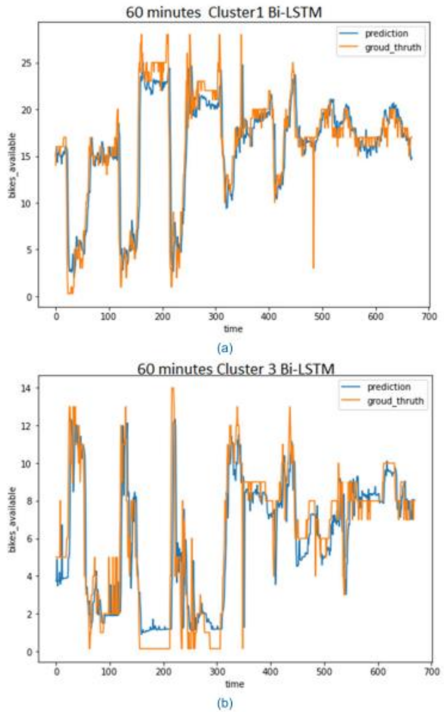
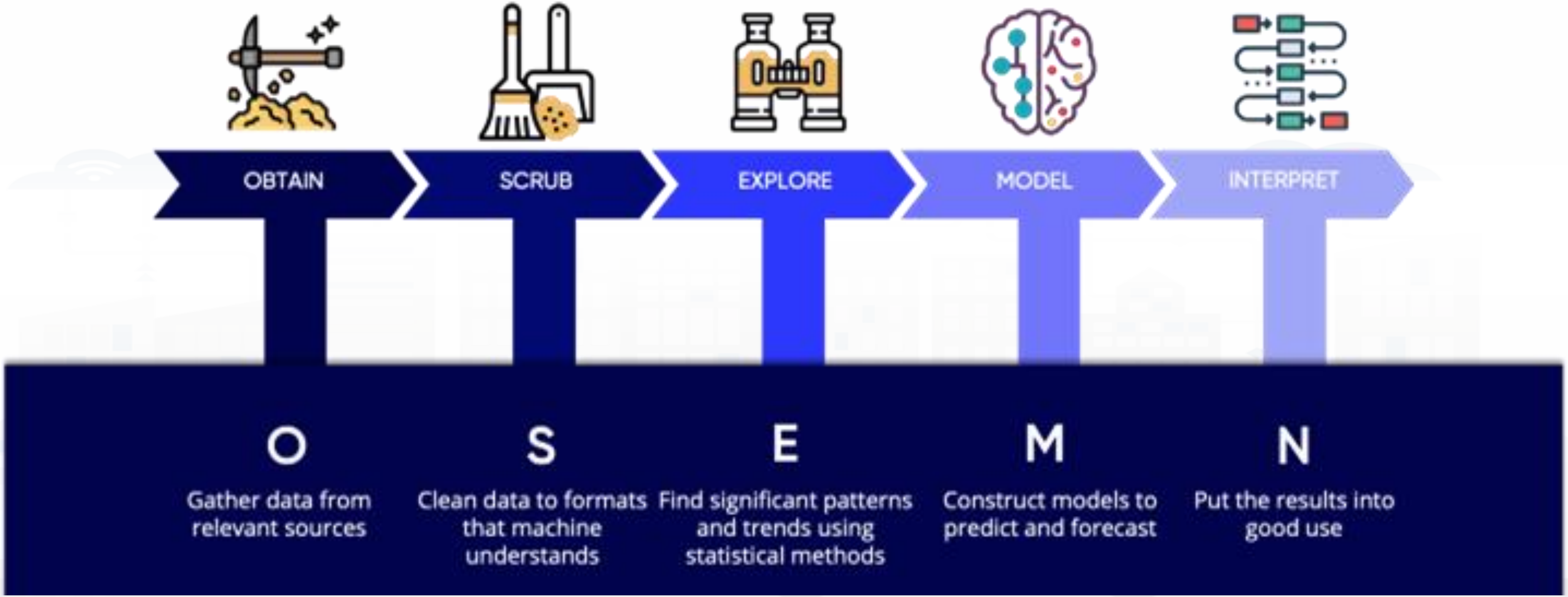


Image Source: https://en.wikipedia.org/wiki/Granger_causality

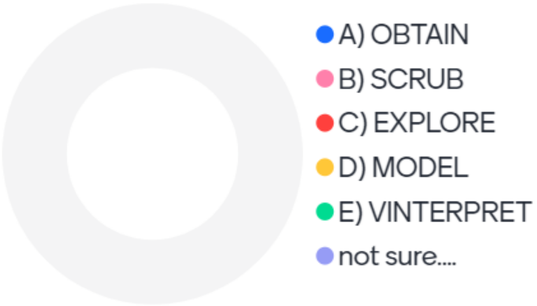
C) Forecast Future Values of a Time-Series



Stepladder to conduct a great time series analysis



In which part of the OSEMN pipeline, data analysts will spend the majority of their time?



Obtain Data

The very first step of a data science project is straightforward. We obtain the data that we need from available data sources.

- You'll need to:
  - query databases
  - receive data in file formats like
  - gather data via connecting via Web
  - generate Synthetic Data to work on

https://www.snap4city.org/dashboardSmartCity/view/Baloon.php?iddasboard=MzcxNw==

Scrub Data

After obtaining data, the next immediate thing to do is scrubbing data. This process is for us to “clean” and to filter the data.

Good data is more important than any analysis method

- Go to Actions:
  - Time granularity casting
  - Handling Data missing - Imputation Strategies
  - “3” -> 3 string numbers??

Scrub Data - Completeness

Information Quality Pillars / **Complete Data**:

- Are there any gaps in the data referring to the period selected from what was expected and on what was actually there

S.AgostinoBikeRack .XLSX

File Modifica Visualizza Inserisci Formato Dati Strumenti Guida Ultima modifica: 2 minuti fa

K22 fx 1

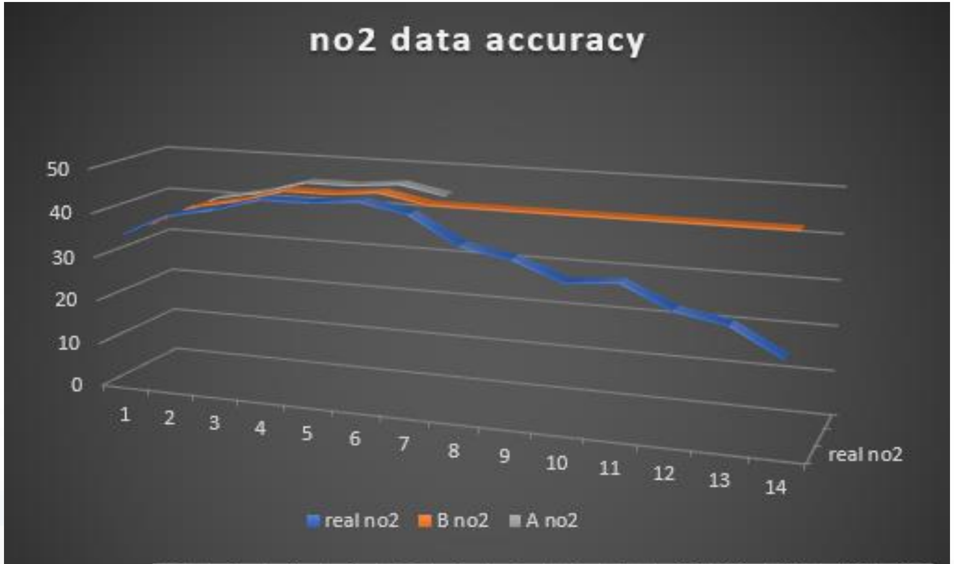
	A	B	C	D	E	F	G	H	I	J	K	L
1			bike-sharing rack			Weather Data			Fefatures Engineered			
2		DateTime	freeStalls	brokenBikes	availableBikes	Temperature	Humidity	Pressure	rain	dP	dS	PwAB
3	0	2019-12-23 00:15:00	6	0	3	12,46	87	997	0	3	3	3
4	1	2019-12-23 00:30:00	6	0	3	12,46	87	997	0	3	6	3
5	2	2019-12-23 00:45:00	3	0	6	null	null	null	0	3	6	6
6	3	2019-12-23 01:00:00	3	0	6	12,12	87	997	0	6	3	6
7	4	2019-12-23 01:15:00	null	null	null	12,14	76	998	0	6	3	3
8	5	2019-12-23 01:30:00	6	0	3	12,14	76	998	0	3	3	3
9	6	2019-12-23 01:45:00	6	0	3	12,14	76	998	0	3	3	3

Scrub Data - Accuracy

Information Quality Pillars / **Accurate Data**:
- are the collected data correct / do they accurately represent what it should

Data Acquisition…
- IoT environment sensor with air pollutants breaks.

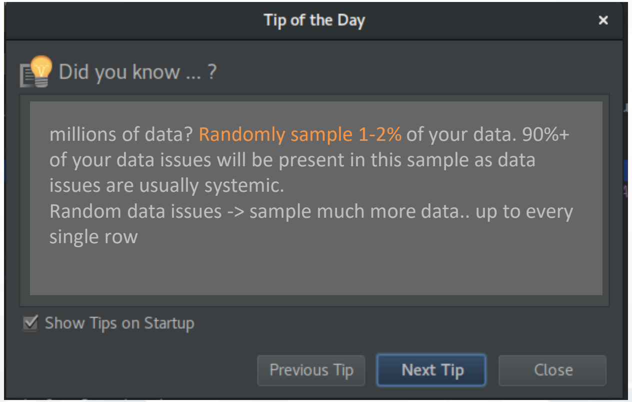
A) keep sending the last value
B) sends the data only if available



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
real no2	35	40	42	45	45	46	44	38	36	32	33	28	26	20
B no2	35	40	42	45	45	46	44	44	44	44	44	44	44	44
A no2	35	40	42	45	45	46	44							

Scrub Data - Information Quality Pillars

- **Validity:** data really measure what is intended?
- **Timely**: data should be received in order and depending on the application really fast!

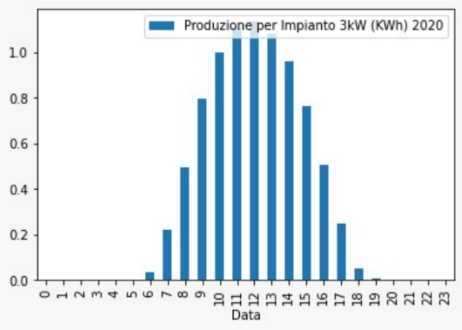


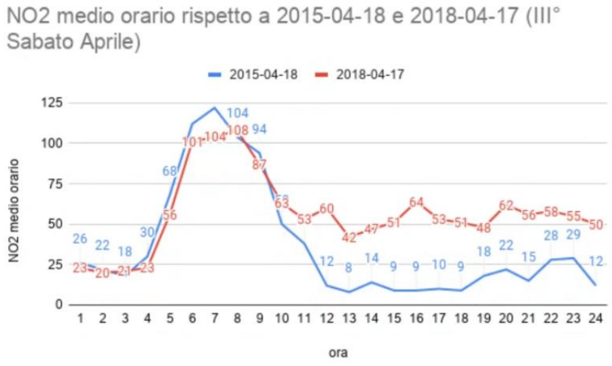
Explore Data

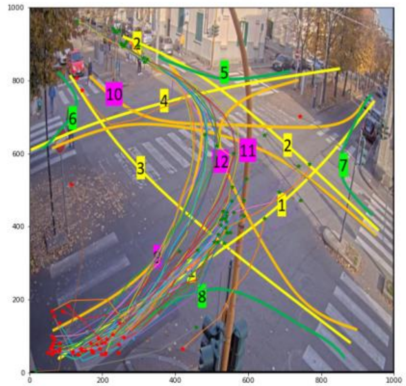
Once your data is ready to be used, and right before you jump into AI and Machine Learning, you will have to examine the data.

-> Does your data meet the assumptions of your intended analysis type

- Distributions
- Patterns / Trends
- Clustering







Model

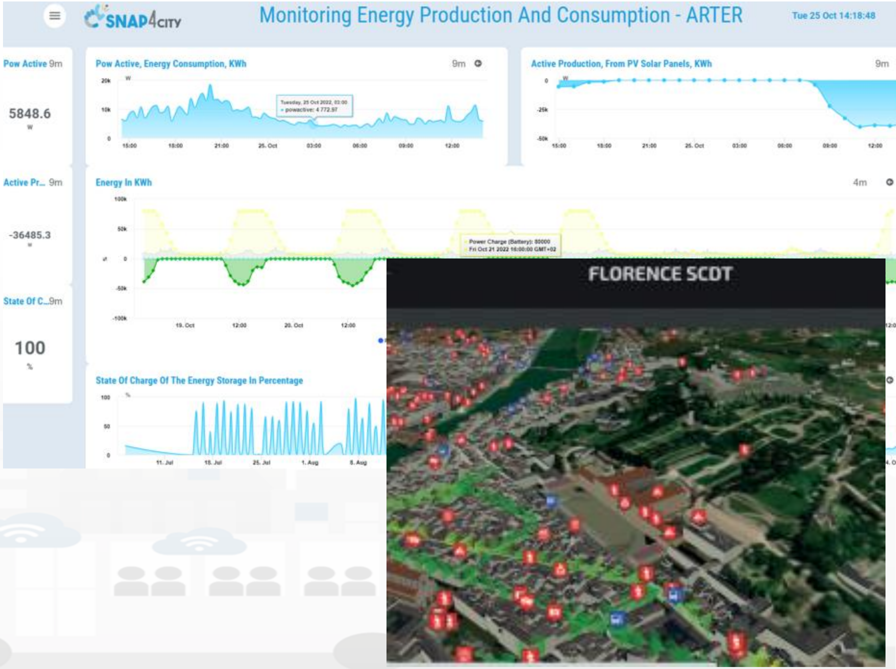
Fortunately there are two main kinds of analysis:

- **Classification Problems**
  - Focus on putting one data record into one of a set of groups
- **Regression** Problems
  - Based on the values recorded predict the value of some other variable of interest

Provide one example of a classification problem and one of a Prediciton problem based on time-series data in the context of Smart Cities

Interpret

- Finally using visualization and other techniques we will interpret the results.
  - Monitoring Dashboards
  - What-if-analysis tools
  - Web/Mobile Application
  - Edge device implementation
  - Early warning systems



Time Series Basics

– What is time series Data?
– What can you do with time series analysis (TSA)
– Stepladder to conduct a great time series analysis… with examples

Road to Time Series Forecasting

- Time Series Characteristics
  - Mathematical formulation of Time Series
  - Autocorrelation
  - Seasonality
  - Stationarity

Forecasting Methods Selection

Mathematical Formulation of Time Series

Time Series is the set of several observations of a phenomenon with respect to time.

The observed phenomenon, called a **variable** Y , can be observed at given instants of time and it can be denoted with Y_t with $t = \{1, 2, 3, ..., T\}$ the time instant.

So a Time Series can be defined as follows: $Y = \{Y_1, Y_2, ..., Y_T\}$

For example, if one were to survey quarterly GDP in millions of euros at chain-linked values (reference year: 2000; raw data) from Q1 1981 to Q2 2008, one would have 110 observations, including:

Y_1 : GDP at the end of Q1 1981 (193,505);
Y_{12} : GDP at the end of Q4 1983 (215,584);
Y_{55} : GDP at the end of Q3 1994 (263,660).

Moments of Time Series can be defined as:

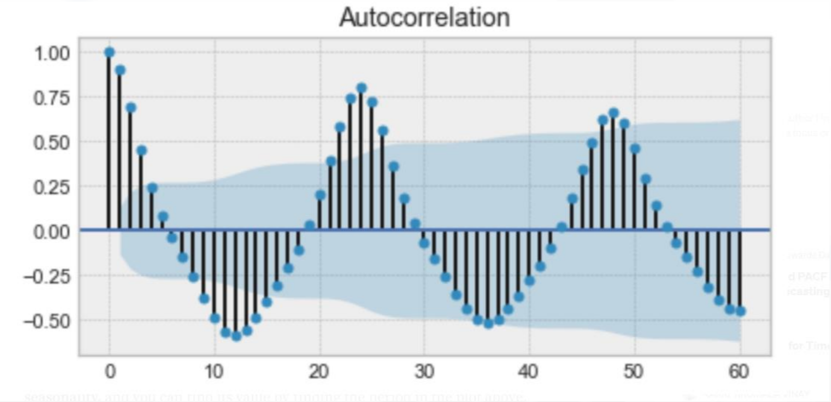
Mean $\mu_t = \mathbb{E}[Y_t]$
Variance $\sigma_t^2 = \mathbb{E}[(Y_t - \mu_t)^2]$
Autocovariance: $\gamma_{t,s} = \mathbb{E}[(Y_t - \mu_t)(Y_s - \mu_s)]$

$\mathbb{E}[X] = \sum_{i=1}^{\infty} x_i p_i$. with Discrete uniform distribution

$$p_i = \frac{1}{T}$$

Time Series Characteristics

Autocorrelation is the similarity between observations as a function of the time lag between them.



Autocorrelation Function Plot (**ACF**)

- The first value and the 24th value have a high autocorrelation. Similarly, the 12th and 36th observations are highly correlated. This means that we will find a very similar value at every 24 units of time.

Notice how the plot looks like sinusoidal function. This is a hint for seasonality, and you can find its value by finding the period in the plot above, which would give 24h

Understanding ACF Plots

We defined a Time Series as follows: $Y = \{Y_1, Y_2, ..., Y_T\}$

Let's now consider the delayed Time Series in a new variable $Z = Y_{t-k}$

Where k is the size of the lag. Setting $k = 3$,

if Y_a is the Italian GDP of 2007,

Z_a is the Italian GDP of 2004.

- To construct a correlogram, the correlations between the historical series and several lagged series of k periods are examined; for example, given the series. $Y_1, Y_2, Y_3, \ldots, Y_{T-2}, Y_{T-1}, Y_T$
- One ideally constructs a table like the following, where K indicates the maximum value of k:
- And the K correlations between the Yt-column and each of the Yt-k columns are examined.

Y_t	Y_{t-1}	Y_{t-2}	Y_{t-3}	$\ldots$	Y_{t-K}
Y_1					
Y_2	Y_1				
Y_3	Y_2	Y_1			
Y_4	Y_3	Y_2	Y_1		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Y_{T-2}	Y_{T-3}	Y_{T-4}	Y_{T-5}	$\vdots$	Y_{T-K-2}
Y_{T-1}	Y_{T-2}	Y_{T-3}	Y_{T-4}	$\vdots$	Y_{T-K-1}
Y_T	Y_{T-1}	Y_{T-2}	Y_{T-3}	$\vdots$	Y_{T-K}

Understanding ACF Plots

- To construct a correlogram, the correlations between the historical series and several lagged series of k periods are examined; for example, given the series. $Y_1, Y_2, Y_3, \ldots, Y_{T-2}, Y_{T-1}, Y_T$
- One ideally constructs a table like the following, where K indicates the maximum value of k:
- And the K correlations between the Yt-column and each of the Yt-k columns are examined.

Y_t	Y_{t-1}	Y_{t-2}	Y_{t-3}	$\ldots$	Y_{t-K}
Y_1					
Y_2	Y_1				
Y_3	Y_2	Y_1			
Y_4	Y_3	Y_2	Y_1		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Y_{T-2}	Y_{T-3}	Y_{T-4}	Y_{T-5}	$\vdots$	Y_{T-K-2}
Y_{T-1}	Y_{T-2}	Y_{T-3}	Y_{T-4}	$\vdots$	Y_{T-K-1}
Y_T	Y_{T-1}	Y_{T-2}	Y_{T-3}	$\vdots$	Y_{T-K}

The calculation is done by varying k from 1 to K and noting the correlation r between the column Y_t and the lagged variable column Y_{t-k}:

$$r_k = \frac{\sum_{t=K+1}^{T}(Y_t - \bar{Y})(Y_{t-k} - \bar{Y})}{\sum_{t=K+1}^{T}(Y_t - \bar{Y})^2}$$

The autocovariance divided by the product of the standard deviations, i.e. the variance

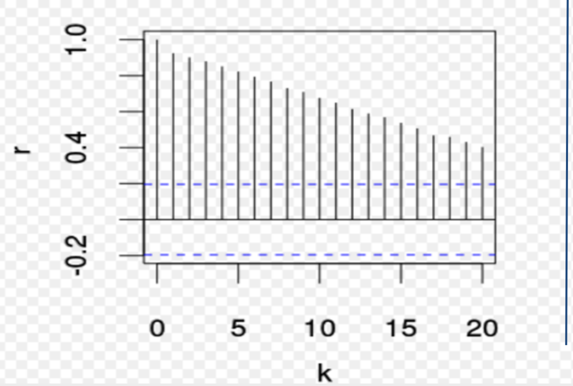
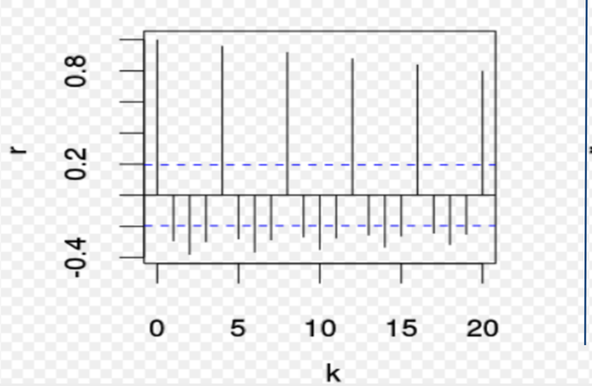
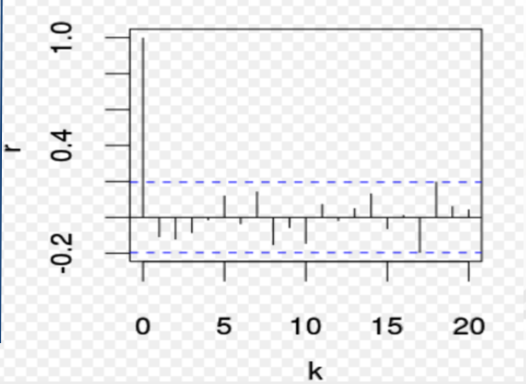
Understanding ACF Plots

The calculation is done by varying k from 1 to K and noting the correlation r between the column Y_t and the lagged variable column Y_{t-k} :

$$r_k = \frac{\sum_{t=K+1}^{T}(Y_t - \bar{Y})(Y_{t-k} - \bar{Y})}{\sum_{t=K+1}^{T}(Y_t - \bar{Y})^2}$$

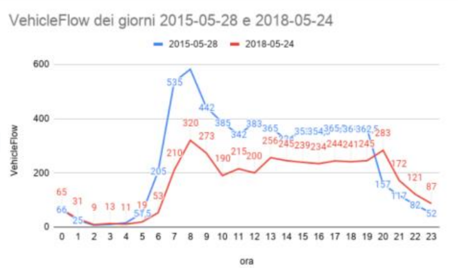
The autocovariance divided by the product of the standard deviations, i.e. the variance

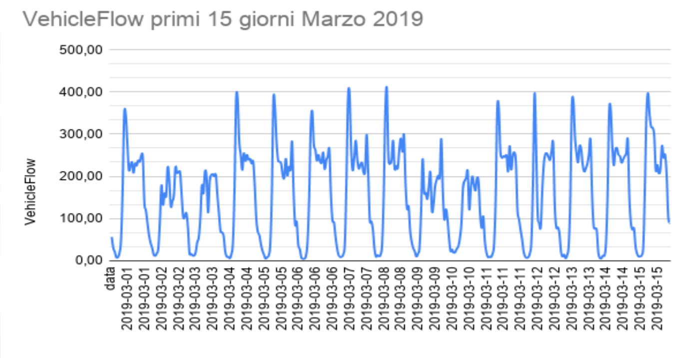
Original Time Series characteristics

linear trend Behaviour	Seasonal component	Stochastic
		

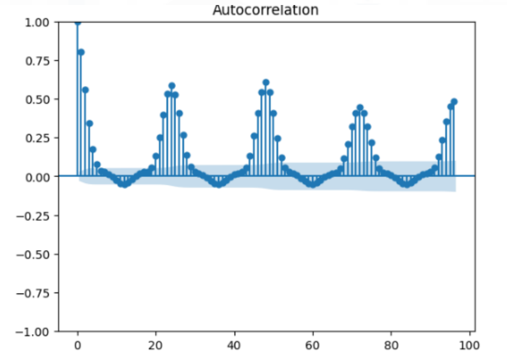
Time Series Characteristics

Seasonality refers to periodic fluctuations. For example, VehicleFlow is high during the day and low during night





Remember that seasonality can also be derived from an autocorrelation plot if it has a sinusoidal shape. Simply look at the period, and it gives the length of the season.

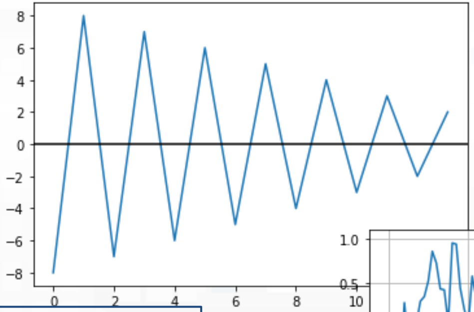


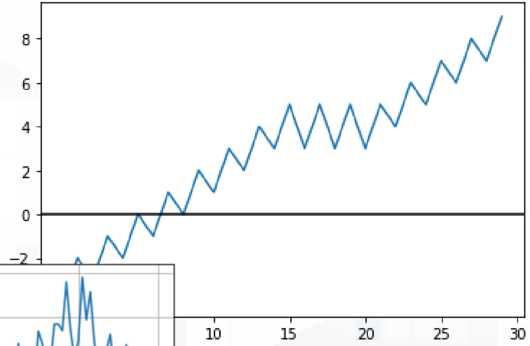
Time Series Characteristics

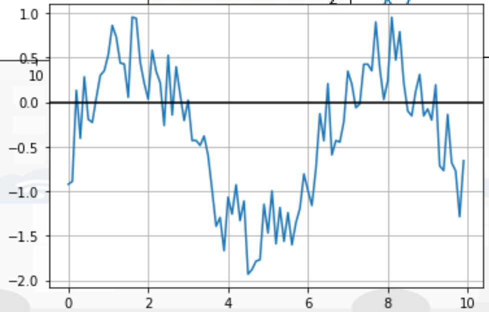
Stationarity is an important characteristic of time series that the majority of statistical forecasting techniques require. A time series is said to be stationary if its statistical properties do not change over time and there is not seasonality…

3 requirements:

- μ const
- σ^2 const
- No Seasonality







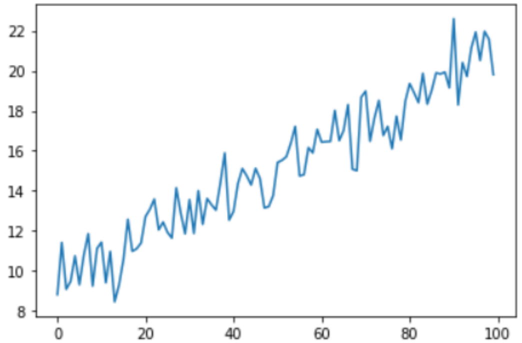
How to check for stationarity:

1) Visually as we did
2) Global mean vs local mean ….
3) **Statistical Tests**

Making a Time Series Stationary

```
T = 100
mean = 0
std = 1

eps = np.random.normal(mean, std, size=T)
b0 = random.random()*10
b1 = random.random()
y = []
for t in range(T):
    yt = b0 + b1*t + eps[t]
    y.append(yt)
plt.plot(y)
```

```
[<matplotlib.lines.Line2D at 0x7faac0353cd0>]
```



$$Y_t = \underbrace{\beta_0 + \beta_1 t}_{\text{straight line}} + \underbrace{\epsilon_t}_{\text{white noise error N(0,k)}}$$

- ! not stationary but somewhat seems predictable...
- Lets define $D_t = Y_t - Y_{t-1} =$

$$\beta_0 + \beta_1 t + \epsilon t - \beta_0 - \beta_1 (t-1) + \epsilon_{t-1} =$$

$$\beta_1 (t - t - 1) + \epsilon_t - \epsilon_{t-1} =$$

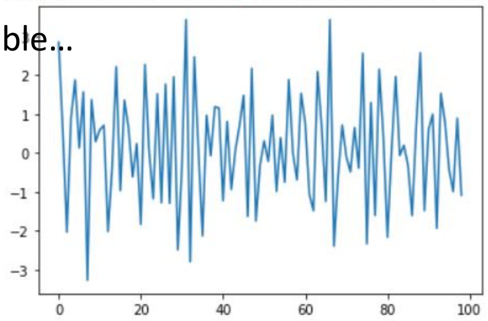
$$\underbrace{b_1 + (\epsilon_t - \epsilon_{t-1})}_{\text{const} \quad \text{indip vars}}$$

μ const b_1

σ^2 k^2 k^2 = 2 k^2 const

```
D = []
for t in range(1,T):
    D.append(Y[t]- Y[t-1])
plt.plot(D)
print("mean {}, variance {}".format(np.mean(D), np.var(D)))
```

```
mean 0.15033575645472239, variance 2.1557327920738136
```



Transformations such as logarithms can help to stabilise the variance of a time series. **Differencing** can help stabilise the mean of a time series by removing changes in the level of a time series, and therefore eliminating (or reducing) trend and seasonality.

Stationary Time-Series tests

- Stationary time series exhibit statistical properties that remain constant over time. This means that the averages, variances and other measures of distribution of the data do not change significantly over time.

- Many statistical models for forecasting time series, such as AutoRegressive Integrated Moving Average (ARIMA) models, assume stationarity of the series. The absence of stationarity can lead to ineffective models or inaccurate forecasts.

Main tests [edit]

Other popular tests include:

- augmented Dickey–Fuller test[2]

  this is valid in large samples.

- Phillips–Perron test
- KPSS test

  here the null hypothesis is trend stationarity rather than the presence of a unit root.

- ADF-GLS test

Unit root tests are closely linked to serial correlation tests. However, while all processes w will have a unit root. Popular serial correlation tests include:

- Breusch–Godfrey test
- Ljung–Box test
- Durbin–Watson test

Forecasting Methods Selection

Characteristics	Statistical Forecasting Techniques	AI prediction models
stationary time-series	Y	Y
non-stationary time-series	N	Y
short-term predictions	well suited	well suited
long-term predictions	applicable	well suited
require a lot of data	often not	Y

Road to Time Series Forecasting

– Time Series Characteristics
- Mathematical formulation of Time Series
- Autocorrelation
- Seasonality
- Stationarity

Forecasting Methods Selection

Examples of Time Series Analysis

Deep learning for short-term prediction of available bikes on bike-sharing stations
E Collini, P Nesi, G Pantaleo
IEEE Access 9, 124337-124347

Long Term Predictions of NO_2 Average Values via Deep Learning
P Bellini, S Bilotta, D Cenni, E Collini, P Nesi, G Pantaleo, M Paolucci
Computational Science and Its Applications–ICCSA 2021: 21st International …

Short-term prediction of city traffic flow via convolutional deep learning
S Bilotta, E Collini, P Nesi, G Pantaleo
IEEE Access 10, 113086-113099

with deepenings on

- Time- series data acquisition with Snap4City
- Check on information data quality pillars
- Data Imputation

- Time-series feature selection for better AI models
- Time-series in good use -> Monitoring Dashboard examp

- Data missing robustness

Short-Term Prediction of Bikes Availability on Bike-Sharing Stations

Bike Sharing

- **Pros**:
  - Eco-friendly
  - Prevent traffic congestions
  - Reduce the probability of social contacts in public transports
  - Regular bikes or e-bikes
- **Problems**:
  - Irregular distribution of bikes on racks/areas
  - Difficulty of knowing in advance their status with a certain degree of confidence
    - available bikes at a specific bike-station
    - free slot for leaving the rented bike

⍰ providing **PREDICTIONS** can be useful to improve quality of service

GOALS

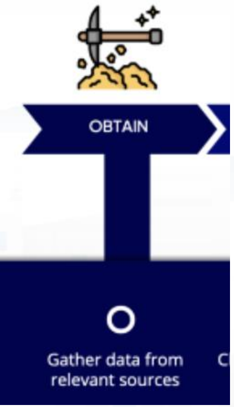
- Producing short-term 1h predictions of:
  (i) number of bikes available in bike-sharing systems stations,
  (ii) free slots.
- Identify the best solution among different AI/ML Techniques.
- Understand which are the most relevant features for the predictive model

Scenario

- The solution and its validation have been performed by using data collected in bike-stations
  - in the cities of **Siena** and **Pisa (Tuscany, Italy)**,
  - in the context of **Sii-Mobility National Research Project on Mobility and Transport**
  - **exploiting Snap4City Smart City IoT infrastructure**

- The data exploited referred to 15 stations in Siena and 24 in Pisa.
  - the status of each station is registered every 15 minutes

Time- series data acquisition with Snap4City



https://www.bicincitta.com/

REST API

- number of bikes available
- the total capacity of the rack
- the number of broken bikes

Time- series data acquisition with Snap4City

– number of bikes available
– the total capacity of the rack
– the number of broken bikes

data ingestion

IOT Directory and Devices

define a **IoT Device Model**:
- model_name
- Acquisition Frequency
- Static Attributes
  - -latitude
  - -longitude
  - -total rack capacity
- Dynamic Attributes
  - number of bikes available
  - number of broken bikes

Time- series data acquisition with Snap4City

data ingestion

IOT Directory and Devices

define a **IoT Device Model**:

- model_name
- Acquisition Frequency
- Static Attributes
  - -latitude
  - -longitude
  - -total rack capacity
- Dynamic Attributes
  - number of bikes available
  - number of broken bikes

IOT Applications

Node-RED

Create the Device form model

iotdirectory
new device
from model

IOT Directory and Devices

Or do it by hand with the Snap4City gui

Time- series data acquisition with Snap4City

IOT Applications

Node-RED

Create the Device form model

iotdirectory new device from model

IOT Directory and Devices

Or do it by hand with the Snap4City gui

IOT Directory and Devices

Bring the device json input data format

JSON

IOT Applications

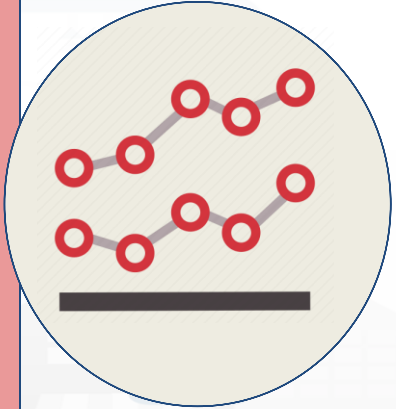
Node-RED

Save the data in the platform

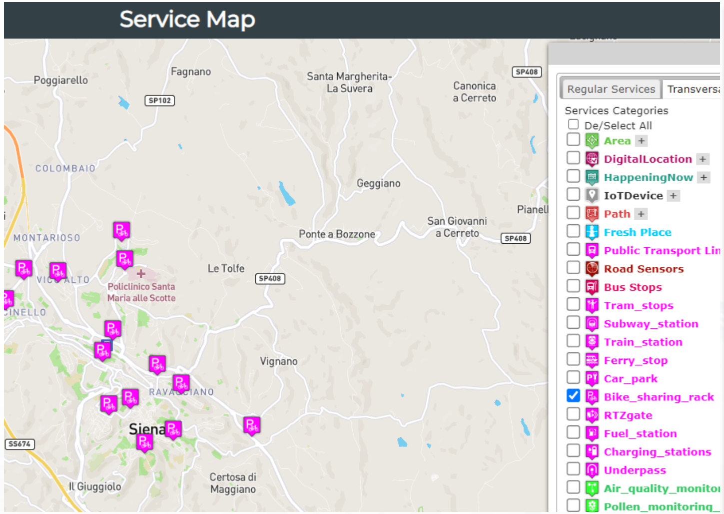
fiware orion out api v2

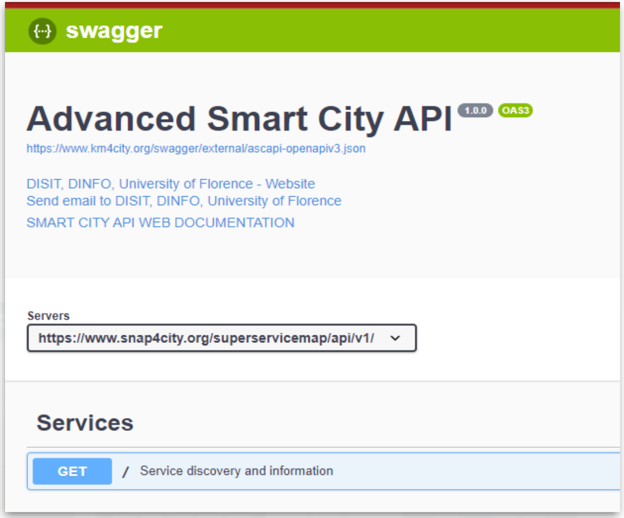
REST API

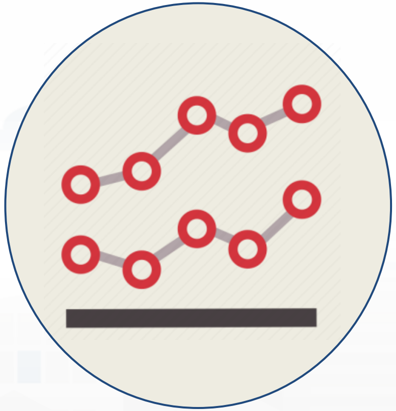
Node-RED



Time- series data acquisition with Snap4City







Data Availability

- The temporal windows of data available for the city of Siena is
  - from June 2019 to January 2020
- The data taken into account for the bike racks of Pisa
  - from December 2019 to March 2020

The data acquired by the stations

- the number of bikes available
- the total capacity of the rack
- the number of broken bikes

Check on information data quality pillars

5 PILLARS of Information Quality:

- **Complete**
- Accurate
- Consistent
- Validity
- Timely

daily?

dati ci sono ma non intero mese

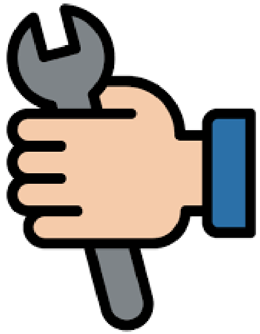
dato quasi mese completo ma c'è un buco

dati ci sono intero mese

	Gen2019	Feb2019	Mar2019	Apr2019	Mag2019	Giu2019	Lug2019	Ago2019	Set2019	Ott2019	Nov2019	Dic2019	Gen2020	Feb2020	Mar2020
PISA	15	0	20	30	31	30	16	0	0	0	0	19	28	29	31
SIENA	15	0	3	0	0	6	22	31	30	31	30	30	28	29	31

Data Imputation

- **Data missing is an inevitable problem** when dealing with real world IoT senso networks.
- Sensors may suffer of problems such a detector malfunction and communication failure.
- Or there could be problems in the dat acquisition phase.



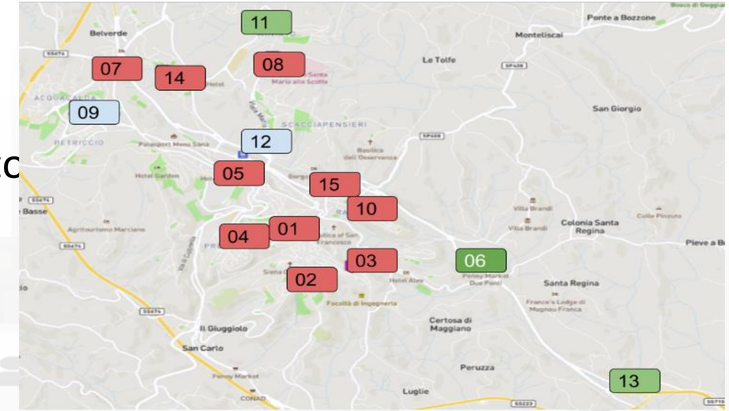
Data Imputation Strategies:

- **Do nothing**
- Imputation using mean/median values
- **Hot Deck Encoding**
- most frequent value
- k nearest neighbours - Mice - Datawig Deep learning based imputation solution

Clustering

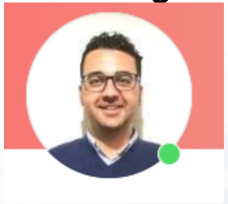
- A **clustering** approach has been applied in order to classify Pisa and Siena stations based on their mean trend H24 of bikes availability
  - This is also correlated to the typical services in the neighborhoods

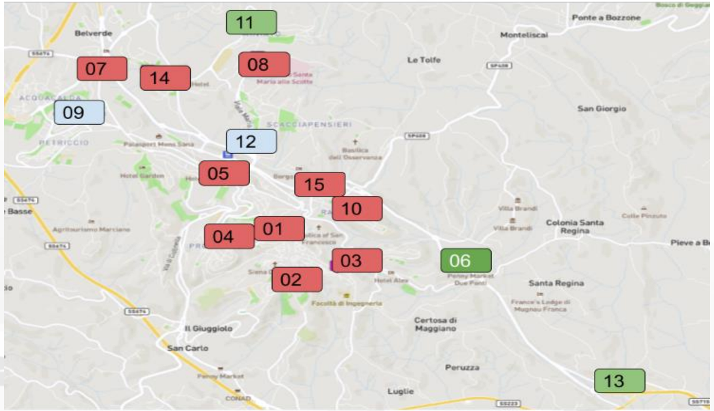
- **K-means** clustering method has been applied to identify clusters
  - The optimal number of clusters resulted to be equal to **3**, and it has been identified by using the **Elbow criteria**



Clustering

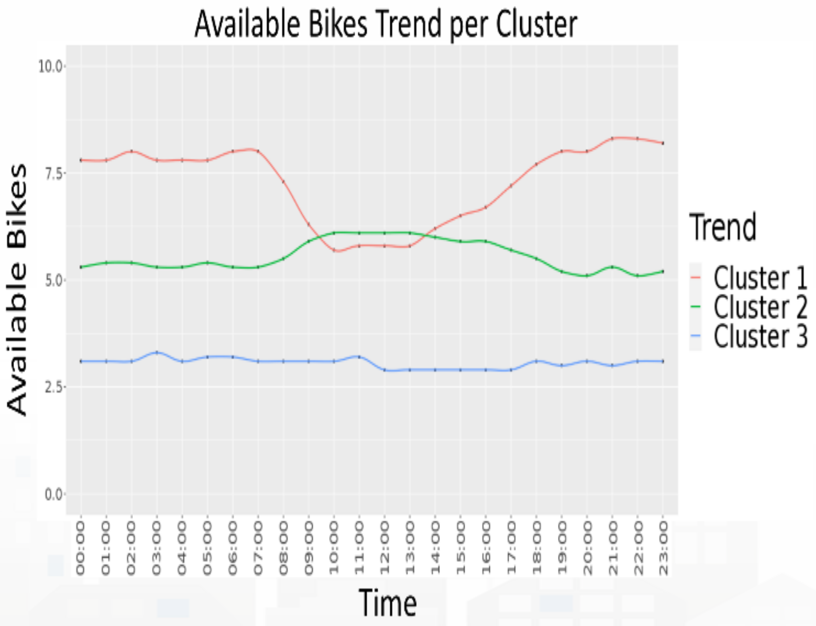
- Descriptive Statistics
- Trend Plots Analysis
- **Clustering**





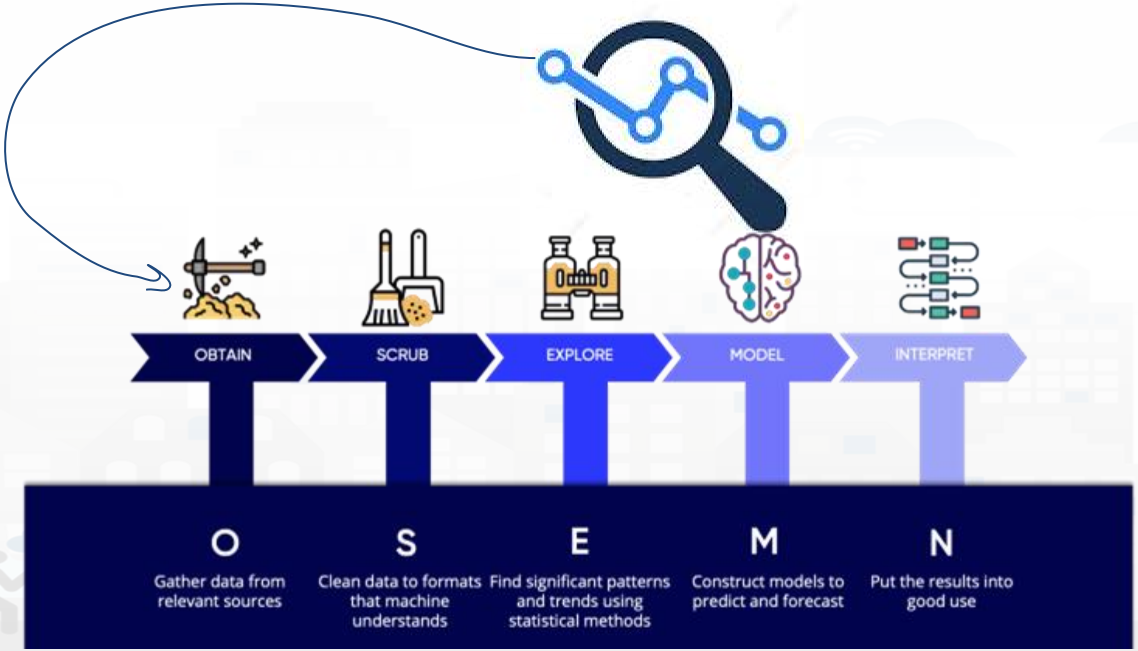
Clustering

- **Cluster 1:**
  - characterized by a decrement of bike availability at lunchtime,
  - Typically located close to the railway stations, airport, etc.
- **Cluster 2:**
  - characterized by an increment of the availability of bikes in the central part of the day (lunch hours, since most of the people are parking their bikes to get lunch).
  - Typically positioned in the central area of the cities,
- **Cluster 3:**
  - almost uniform trend in the bike availability
  - mainly positioned in the peripheral areas of the city



Modelling Phase

State of the Art Analysis of AI architectures & **<u>data sources</u>** used for the prediction



Modelling Phase

TABLE I

COMPARISON OF RELATED WORK SOLUTIONS, WITH MAIN ATTENTION TO DEEP LEARNING ASPECTS AND BETTER RESULTS.

citation	Target	Features	Dataset	Model	Reported Best Resutls
[25]	1h, 2h, 3h bike rentals and returns	Bike rented, Bike returned, Avg temperature, Wind speed, Sky cover, Rain, holiday or Sunday, time, weekday, month, year	ThessBike	RF, XGBoost, GB, DNN	RF / Rentals / returns MAE / 0.85 / 0.82 MSE / 2.77 / 2.76 RMSLE / 0.46 / 0.46 R2 / 0.64 / 0.63
[24]	Hourly Bike number change in station	Usage features, spatial features, temporal features	Citi Bike dataset July – August 2017	XGBoost tree, RF, DNN	XGBoost tree MAE / 1.8159 AP / 0.7085
[26]	1h rental bikes rented	Rental bikes rented, Weekend/weekday, Day of the week, Holidays, Functional/non functional, Temperature, Humidity , Windspeed, Visibility, Dew Point, temperature, Rainfall, snowfall	Seoul (South Korea)	RF, SVM, k-Nearest neighbours (KNN), Classification and Regression Trees (CART)	RF results: R2 / 0.88 RMSE / 216.01 MAE / 130.52 CV / 30.63 PI / 0.73
[27]	Hourly rental bike demand	Temperature, Humidity, Windspeed, Visibility, Dewpoint, Solar radiation, Snowfall, Rainfall, number of bikes rented per hour, date information.	Seoul (South Korea	LR, XGBoost, SVM, Boosted Trees, XGBoost Trees	XGBoost results: R2 / 0.92 RMSE / 174.68 MAE / 109.89 CV / 24.92
[28]	Long terms predictions	Timestamp, count of new bike shared, temperature, humidity, windspeed, weather code, is holiday, is weekend, season	London	LR, RF, XGBoost, SVM, AB, BGR	RF results: MAE / 0.04 MSE / 0.01 RMSLE / 0.03 R2 / 0.95
[23]	1h number of rider...	Number of riders, Season, year, month, hour, day, holiday, weekday, working, weather...	Rental Company...	DNN	80% accuracy

Features

categories	Metrics	Description of metric variable
BASELINE - HISTORICAL	AvailableBikes	The number of bikes available
	Time, week, month, day	Time of the day of the data, month and week of the year and day of the year
	Day of the week	The day of the week 1,..., 7
	Weekend, holiday	1 if Saturday or Sunday , 0 otherwise 1 if the day is a holiday, 0 otherwise
	Previous week, previous day	The previous week of the year and the previous day of the year

Features

categories	Metrics	Description of metric variable
REAL-TIME WEATHER AND WEATHER FORECAST	Max Temperature, Min Temperature, Temperature	Temperature values
	Humidity	The humidity of the hour prior to the observation measurement in percentage
	Rain	ml of rain registered in the hour prior to the observation measurement
	Pressure	Pressure in mb
	WindSpeed	Average wind speed registered in the hour prior to the observation measurement in km/h
	Cloud Cover Percentage	Cloud Cover expressed in percentage
	Sunrise	Hour of the sunrise

Features

categories	Metrics	Description of metric variable
DIFF FROM ACTUAL VALUES AND PREV. OBSERVATIONS	dPweek	Previous observation’s difference of the previous week
	dSweek	Subsequent observation’s difference of the previous week
	dPDay	Previous observation’s difference of the previous day
	dSDay	Subsequent observation’s difference of the previous day
	dP2weeks	Previous observation’s difference between the previous week and two weeks earlier
	dS2weeks	Subsequent observation’s difference between the previous week and two weeks earlier

Features

categories	Metrics	Description of metric variable
DIFF FROM ACTUAL VALUES AND PREV. OBSERVATIONS	dPweek	evious week
	dSweek	previous week
	dPDay	evious day
	dSDay	previous day
	dP2weeks	the previous
	dS2weeks	en the previous

the difference between the number of available bikes in the observation day (D) at the time slot t and the number of bikes during the **Previous** time slot (t-1) of the previous day (D-1).

$$dPDay = availableBikes_{D,t} - availableBikes_{D-1,t-1}$$

Predictive AI architecture Analysis

- With a temporal target of 1h, which is the most critical short-term prediction slot ensemble learning techniques such as **Random Forest** (RF) and **Extreme Gradient Boosting Machines** (XGBOOST) are powerful techniques that must be considered for this type of problem.
- It has also been taken into consideration deep learning solutions such as **DNN** architecture with **LSTM** and based on the results of the related works also with a **Deep Bidirectional-LSTM (Bi-LSTM)** Neural Network

Evaluation Metrics

Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(obs_i - pred_i)^2}{n}}$$

R-Squared(R2)

- $\overline{y} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{obs_i}$
- $R^2 = 1 - \left(\frac{\sum_{i=1}^{n}(obs_i - pred_i)^2}{\sum_{i=1}^{n}(obs_i - \overline{y})^2}\right)$

Mean Absolute Scaled Error (MASE)

$$q_t = \frac{obs_t - pred_t}{\frac{1}{n-1}\sum_{i=2}^{n}|obs_i - obs_{i-1}|}$$

$$MASE = mean\,(|q_t|), \qquad t = 1, \dots, n$$

Mean Absolute Error (MAE)

$$MAE = \frac{\sum_{i=1}^{n}|obs_i - pred_i|^2}{n}$$

Deep Learning Models Configuration

The architecture of the Deep Learning neural networks is made up of **4 layers** with specific units of the selected architecture (e.g.: LSTM units for LSTM networks) and optimized hyperparameters via random search.

- The number of neurons for the input layer is equal to 64 or 128;
- for the 2nd layer 64, 32;
- for the 3rd layer 16, 32.
- The last layer has only one neuron with a sigmoid activation function, in order to obtain a value in the range 0, 1 (the input data for the models were normalized using a Min Max scaler).

Deep Learning Models Configuration

- The **batch size** was set to 32 and 64 samples.
- The **dropout rate** for each layer was optimized with the values 0.1, 0.25, 0.5.
- For each model, the **Adam Optimizer** has been chosen with learning rate optimized among 0.05, 0.005, 0.0005 and 0.00005.
- **MSE** was selected as loss function to be monitored during the optimization.
- The number of epochs was set to a maximum value of 1000, because the training strategy used the **Early Stopping** method for determining the optimum epoch number minimizing the RMSE of the validation set, restoring the weights of the best model at the end of the learning process.
- As to LSTMs and Bi-LSTMs inputs were organized through a sliding window with **4 timesteps**, which is equivalent to the values of the previous hour with respect to the prediction time.

Experimental Results

- The data used for this training range from the 16th of December 2019 to the 9th of February 2020. The successive two weeks (10/02/2020 – 23/02/2020) have been used for the validation and the test set includes data from the 24th of February 2020 to the 8th of March 2020
- The machine learning solutions were compared based on the **MAPE** for the prediction targets of 15, 30, 45 and 60 minutes.

Comparative Results	Cluster1:				Cluster2:				Cluster3:			
	15'	**30'**	**45'**	**60'**	**15'**	**30'**	**45'**	**60'**	**15'**	**30'**	**45'**	**60'**
RF	35.16	44.93	53.73	59.57	107.03	146.16	196.55	238.49	30.29	31.60	35.13	36.49
XGBoost	18.75	27.16	40.33	49.09	58.43	83.54	112.46	119.56	28.62	27.30	26.97	29.36
DNN	21.12	28.39	36.01	49.56	109.69	127.23	149.84	178.23	30.29	28.00	27.98	28.68
LSTM	17.68	40.56	44.54	51.16	85.09	120.00	**79.30**	164.00	22.13	**22.91**	26.21	**25.88**
Bi-LSTM	**16.46**	**25.35**	**33.00**	**45.53**	**52.18**	**63.45**	132.00	**92.62**	**21.98**	23.00	**25.15**	27.32

Hyperparameter Details

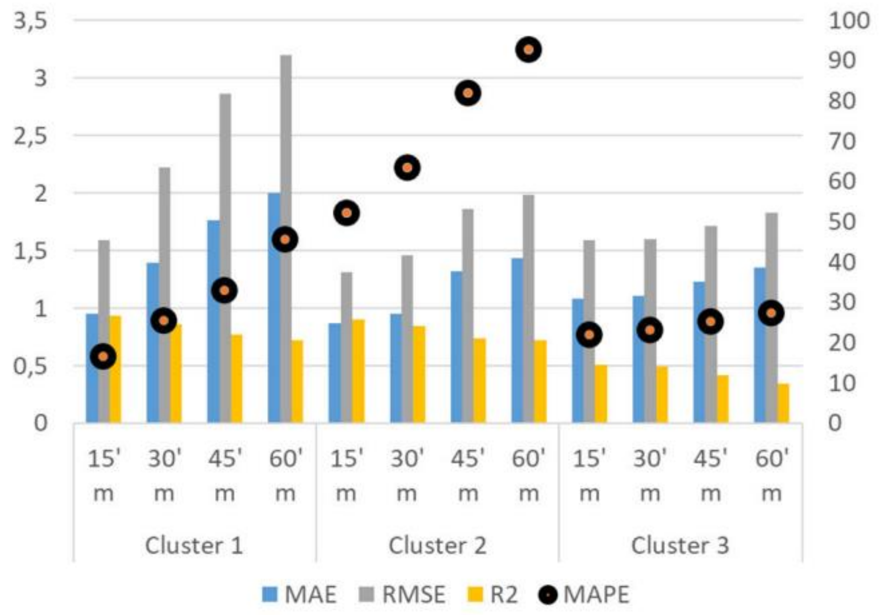
In general, Deep Recurrent Neural Networks architectures outperformed the ensemble learning techniques.

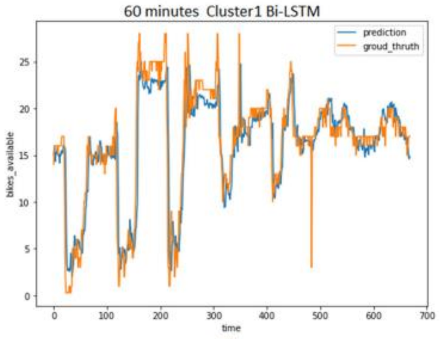
Overall, the best machine learning technique for the prediction of the number of available bikes turned out to be the Bi-LSTM.

The details on the hyperparameters resulting from Random Search Optimization of Bi-LSTM for the temporal target of 60 minutes are reported

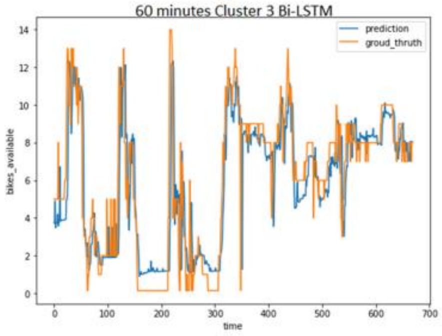
negMSE	Units 1st layer	Units 2nd layer	Units 3rd layer	Dropout Rate	Learning Rate	Batch Dim
Cluster 1						
-0.014	**64**	**32**	**32**	**0.5**	**0.0005**	**32**
-0.016	128	32	32	0.1	0.005	64
-0.44	64	64	16	0.25	0.0005	64
Cluster 2						
-0.011	**64**	**64**	**32**	**0.1**	**0.00005**	**32**
-0.012	64	64	16	0.5	0.0005	32
-0.019	64	32	32	0.1	0.05	64
Cluster 3						
-0.013	**32**	**32**	**16**	**0.0005**	**0.5**	**32**
-0.015	64	32	32	0.005	0.25	64
-0.016	64	64	16	0.00005	0.1	64

Predictions On Representative Sensors





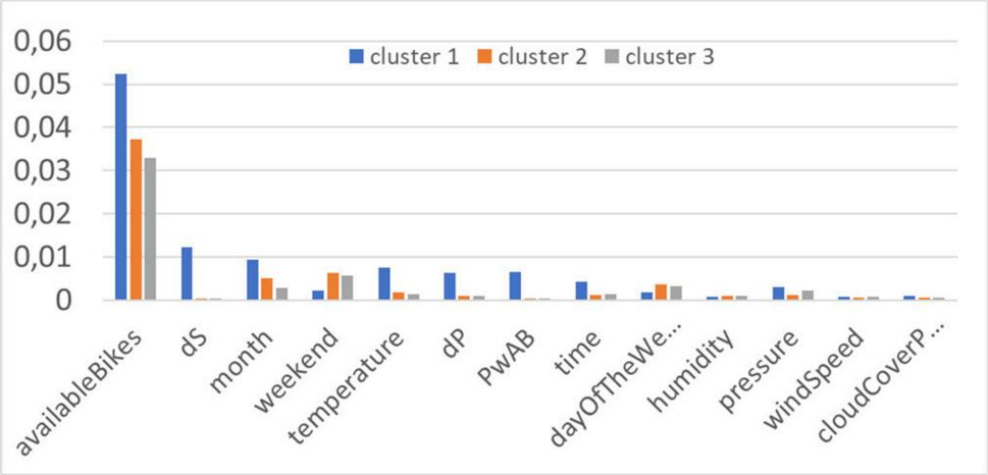
(a)



(b)

Feature Importance Analysis

To evaluate the relevance of features used by Bi-LSTMs for short-term bike availability prediction on the representative bike racks of Pisa and Siena, a SHapley Additive exPlanations (**SHAP**) feature importance analysis was performed



Conclusions

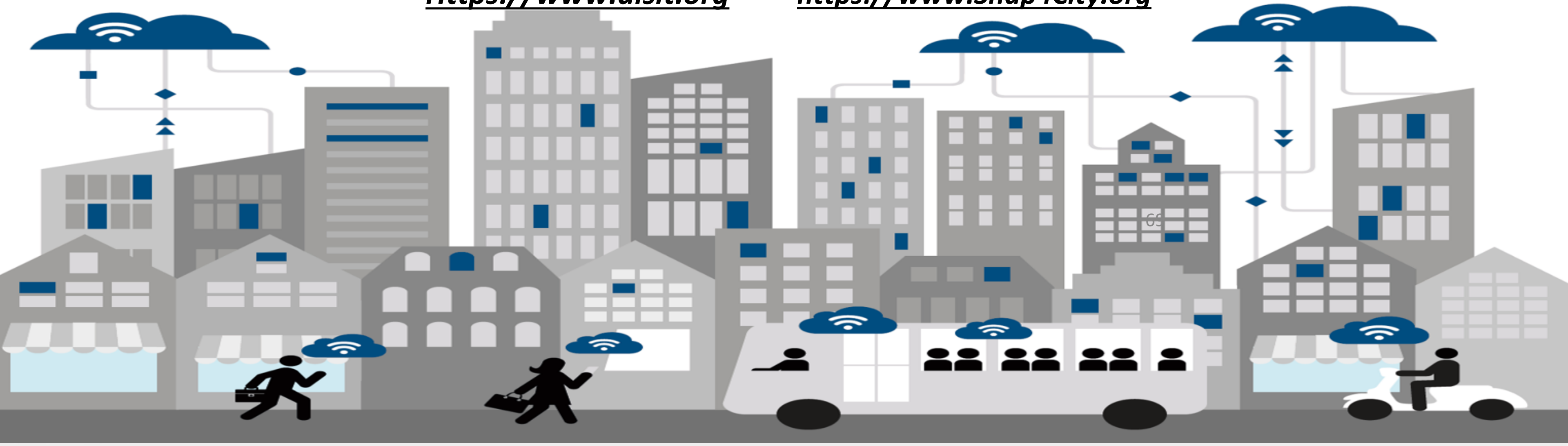
- This work studied machine learning methods to predict **bike availability in bike-sharing systems with smart stations**.
- The proposed method takes high dimensional time-series data from each station and uses real-time and forecast weather information as input to perform long term **predictions 15-30-45-60 minutes**
- The **clustering** process classified bike racks into 3 clusters, where the representative sensors were identified.
- The proposed solution demonstrated that when it comes to short-term prediction the **Bi-LSTM** neural network architecture is the most suitable machine learning technique for this problem.
- The results in terms of **Mean Absolute Error** in the worst-case have achieved an error of **2 bikes** for the 60 minutes prediction on the bike rack.
- The most important feature which has been identified by using **SHAP** analysis **is the number of available bikes in all the clusters**.

Powered by

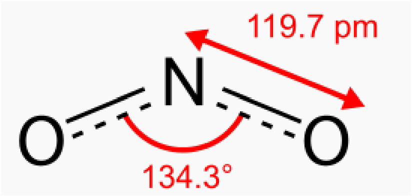
Long Term Predictions of NO2 Average Values via Deep Learning

University of Florence, DISIT Lab, Snap4City

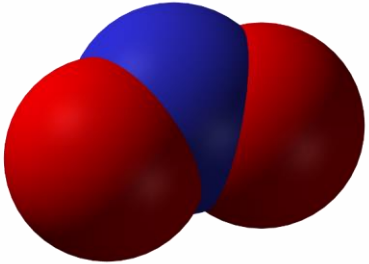
Https://www.disit.org https://www.Snap4City.org



NO2



NITROGEN DIOXIDE



The *European Union* has created a legislative program in which are defined limits on the hourly and yearly concentration of the pollutants (**40 µg/m3 for the yearly mean value of NO2**)

THE CHALLENGES

- The majority of works at the **state-of-the-art** in the Air Quality Prediction are **short-term** based hourly or a few days of the concentrations of the air pollutants.
- In order to make long-term predictions we developed six **Deep Long Short-Term Memory Neural Network** to make the predictions for the temporal taget of **30, 60, 90, 120, 150, 180** days.

The Target

To estimate the **yearly mean value of NO2** it has been used the **progressive mean value** of this pollutant. This value is calculated day after day cumulating the mean daily value and divide this by the number of cumulated days

$$\mathrm{NO_2Cumulated_i} = \sum_{k=1}^{i} \mathrm{NO_2}_k$$

$$\mathrm{NO_2progressiveMean_i} = \frac{\mathrm{NO_2Cumulated_i}}{\mathrm{i}}$$

i-th day
i={1,365}

Iot Sensors used

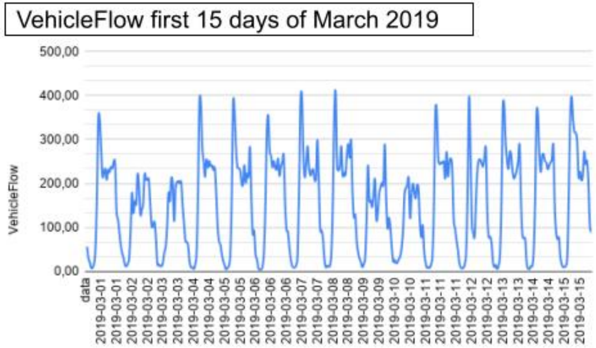
Service Map

More Data

METEOROLOGICAL FEATURES

TEMPORAL INFORMATIONS

NOX

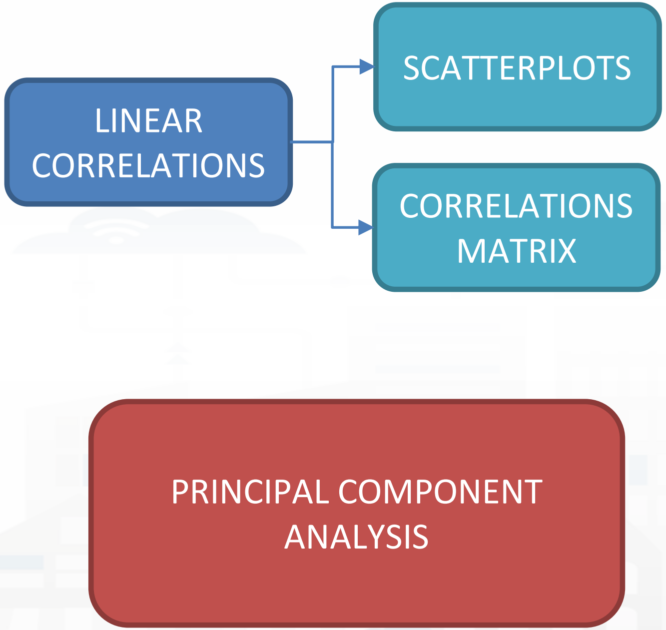


$$NO_x Domestic_i = (K + A * Tmedia_i + B * Tmedia_i^2) * 1000$$

where: $K = 2.224884427$, $A = -0.148281912$, $B = 0.00276720916887206$

Metric	Details
Date	UTC format of the day of prediction YYYY-MM-DD
Year	Of the observation {2014, …, 2020}
Month	Of the observation {1, …12}
dayOfTheYear	Day number in the year {1, …365/366}
dayOfTheMonth	Day number in the month {1, …31}
dayOfTheWeek	Day of the week {1, …, 7}
weekend	Saturday or sunday 1, 0 otherwis
festivity	Festivity 1, 0 otherwise
workingDay	Not a saturday or sunday and it is not a festivity
ferialDay	1 if the day is not a sunday or a festivity
NO_2	The NO_2 hourly mean of the observation day in $\mu g/m^3$
Tmin	The min temperature of the day in °C
Tmean	The mean temperature of the day in °C
Tmax	The max temperature of the day in °C
dewpoint	The dew point temperature in °C
windMean	The mean value of the wind of the day in km/h
windMax	The max value of the wind of the day in km/h
Humidity	The humidity of the day in %
pressioneSLM	The air pressure in millibar (mb)
NOx	The NOx value of the day in kg
numberOfVehicles	The number of vehicles of the day
NO_2cumulated	The cumulated value of NO_2 up to the day
NO_2progressiveMean	The progressive mean value of NO_2 up to the day
numberOfVehiclesCumulated	The number of vehicles cumulated up to the day
NOxDomesticCumulated	The cumulated value of NOx up to the day
NOxDomesticProgressiveMean	The progressive mean value of NOx up to the day

Feature Identification



- The features used as input for the predictive models are:

- **Month**
- **dayOfTheYear**
- **NO2**
- **Tmean**
- **Humidity**
- **windMean**
- **NoxDomestic**
- **numberOfVehicles**
- **NO2cumulated**
- **NO2progresseveMean**
- **numberOfVehiclesCumulated**

Feature Selection

Why don't we give all the features to the ML algorithm and let it decide which feature is important?

- **Curse of dimensionality**: as the number of features or dimensions grows, the amount of data we need to generalize accurately grows exponentially.
- **Occam's Razor**: We want our models to be simple and explainable. We lose explainability when we have a lot of features.
- **Garbage In Garbage out**: Most of the times, we will have many non-informative features. For Example, Name or ID variables. Poor-quality input will produce Poor-Quality output.

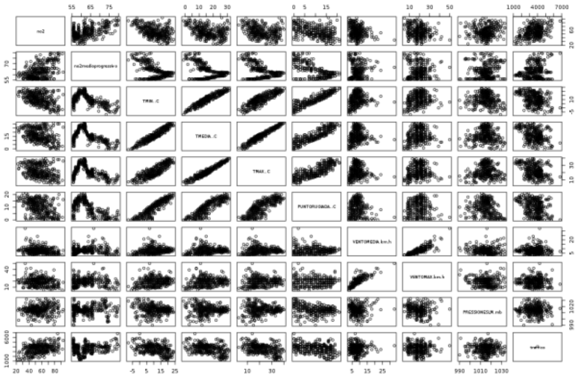
There are plenty of possibilities to conduct a feature selection analysis

- Linear Correlation Analysis
- Principal Component Analysis

Linear Correlation Analysis

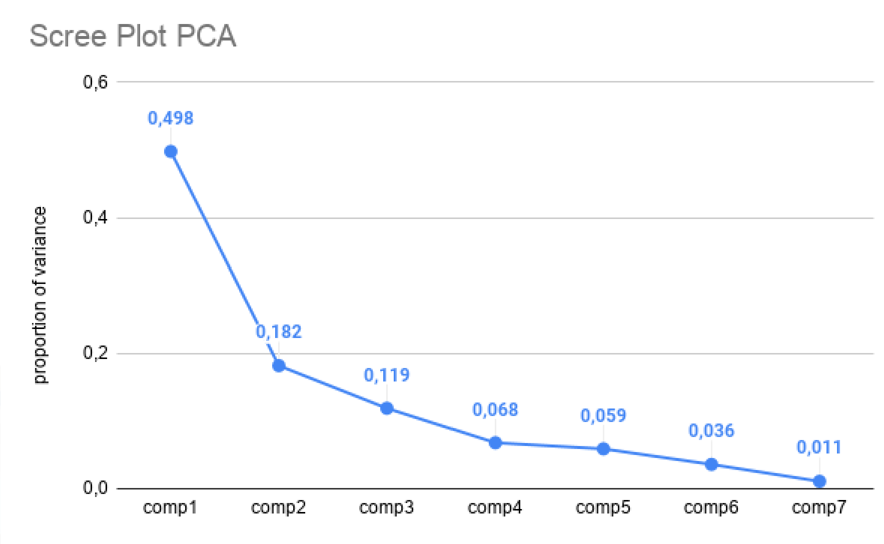
- Through correlation, we can predict one variable from the other.
- The logic behind using correlation for feature selection is that the good variables are highly correlated with the target.
- Otherwise If two variables are correlated, we can predict one from the other. Therefore, if two features are correlated, the model only really needs one of them, as the second one does not add additional information

	NO_2	NO_2medPro	TMIN	TMED	TMAX	PuntRug	VentMed	VentMax	PressSLM	numVeiCum	umidità
NO_2	1	0.494	-0.452	-0.371	-0.297	-0.441	-0.216	-0.200	0.097	0.355	-0.408
NO_2medPro	0.494	1	-0.512	-0.4591	-0.388	-0.584	0.060	0.008	0.102	0.198	-0.192
TMIN	-0.452	-0.512	1	0.945	0.854	0.905	0.162	0.218	-0.147	-0.381	-0.250
TMED	-0.371	-0.459	0.945	1	0.969	0.868	0.092	0.170	-0.022	-0.411	-0.408
TMAX	-0.297	-0.388	0.854	0.969	1	0.802	-0.001	0.109	0.071	-0.420	-0.465
PuntRug	-0.441	-0.584	0.905	0.868	0.802	1	-0.096	0.022	-0.225	-0.344	0.061
VentMed	-0.216	0.060	0.162	0.092	-0.001	-0.0969	1	0.833	-0.046	0.001,	-0.442
VentMax	-0.200	0.008	0.218	0.170	0.109	0.022	0.833	1	-0.138	-0.029	-0.379
PressSLM	0.097	0.102	-0.147	-0.022	0.071	-0.225	-0.046	-0.138	1	-0.010	0.395
numVeiCum	0.355	0.198	-0.381	-0.411	-0.420	-0.344	0.001	-0.029	-0.010	1	0.171
umidità	-0.0408	-0.192	-0.250	-0.408	-0.465	0.061	-0.442	-0.379	0.395	0.171	1



Principal Component Analysis

- PCA is a multivariate data analysis based on projection methods that results in a matrix that summarizes how our variables all relate to one another in different **principal components**.
- data reduction technique that transform the dataset into a compressed form that capture maximum information (**proportion of variance** top principal components)

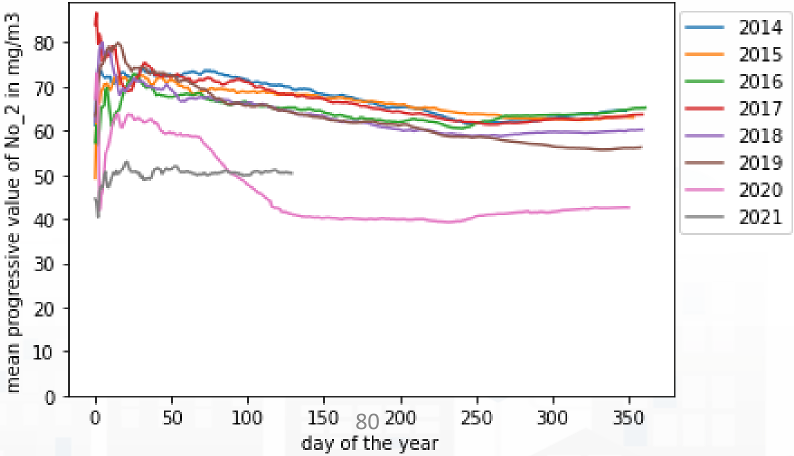


parametro	comp1	comp2	comp3	comp4	comp5
NO_2	0.21492	0.03753	0.21523	0.12079	**0.49583**
NO_2cumulated	-0.29702	**0.33402**	-0.09504	-0.03905	0.03549
NO_2progressiveMean	**0.31897**	-0.25867	0.10213	0.05706	-0.04563
Tmin..C	**-0.30745**	-0.27795	0.06725	0.10825	0.08754
Tmean..C	-0.29595	**-0.31203**	0.16324	0.00393	0.13399
Tmax..C	-0.2687	**-0.31676**	0.24441	-0.06649	0.1375
dewPoint..C	**-0.31326**	-0.15871	0.17945	0.23102	0.04431
windMean.km.h	-0.00725	-0.28206	**-0.6145**	-0.14964	0.07701
windMax.km.h	-0.03454	-0.30142	**-0.59137**	-0.03938	0.09913
humidity	0.01218	**0.43378**	0.04680	-0.42877	-0.07932
pressioneSLM.mb	0.04822	-0.01663	0.18496	**-0.91479**	0.22794
numberOfVehicles	0.14502	0.16311	-0.12736	0.21015	**0.78224**
numberOfVehiclesCumulated	**-0.29235**	0.34455	-0.0991	-0.03161	0.03886
NO_xDomestic	**0.30408**	0.27471	-0.06801	0.00842	-0.13415
NO_xDomesticCumulated	-0.30356	**0.30434**	-0.11701	-0.04954	0.05715
NO_xDomesticProgressiveMean	**0.34165**	-0.1894	0.07221	0.05133	-0.04398

Table 2. Principal Components analysis (a part).

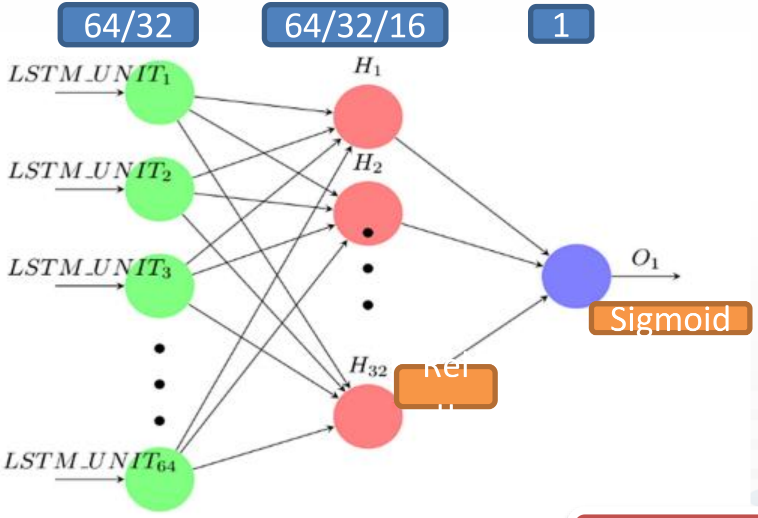
Actual Time Trends

- The data used refers to the years from 2014 to 2020.
- Training set 2014 – 2017
- Test set 2019



Deep LSTM Networks Details

Random Search Hyperparameters Optimization



Mean Squared Error Loss

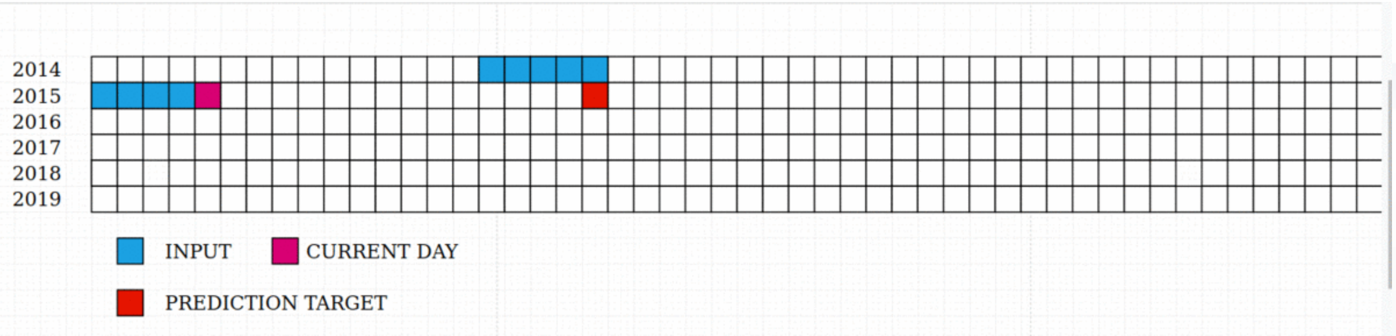
$$L(y, \hat{y}) = \frac{1}{N} \sum_{i=0}^{N} (y - \hat{y}_i)^2$$

Adam Optimizer
Lr between 0,05 0,005 0,0005

Batch size 32 or 64

Early Stopping

- The data have been organized using a sliding window approach made of:
  - The data of 20 days preceding the prediction day
  - The data of 20 days preceding the day of the prediction target of the previous year



(batch_size, timesteps, features)

Evaluation Metrics

Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(obs_i - pred_i)^2}{n}}$$

R-Squared(R2)

- $\overline{y} = \frac{1}{n} \sum_{i=1}^{n} \mathrm{obs_i}$
- $R^2 = 1 - \left(\frac{\sum_{i=1}^{n}(obs_i - pred_i)^2}{\sum_{i=1}^{n}(obs_i - \overline{y})^2}\right)$

Mean Absolute Scaled Error (MASE)

$$q_t = \frac{obs_t - pred_t}{\frac{1}{n-1}\sum_{i=2}^{n}|\,obs_i - \ obs_{i-1}|}$$

$$MASE = mean\,(|q_t|), \qquad t = 1, \ldots, n$$

Mean Absolute Error (MAE)

$$MAE = \frac{\sum_{i=1}^{n}|obs_i - pred_i|^2}{n}$$

Results

metric	model30	model60	model90	model120	model150	model180
MAE	1.21	1.31	1.52	2.04	2.31	2.37
RMSE	2.16	2.61	4.18	6.77	7.83	7.93
MAPE	1.99	2.20	2.65	3.57	4.07	4.18
R2	0.91	0.83	0.80	0.54	0.45	0.14

Table 4. Assessment of the predictive models with respect to the actual values of the 2019.

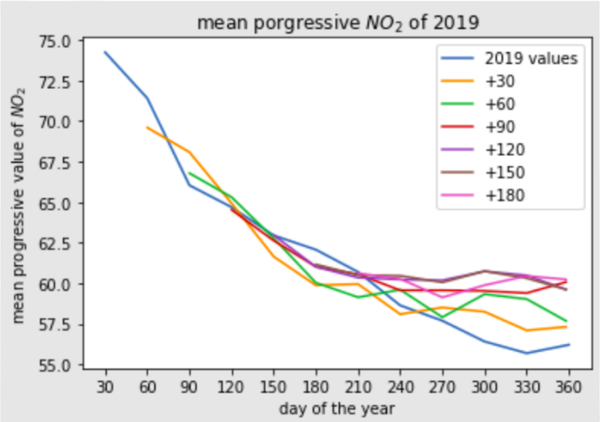
Model Comparison

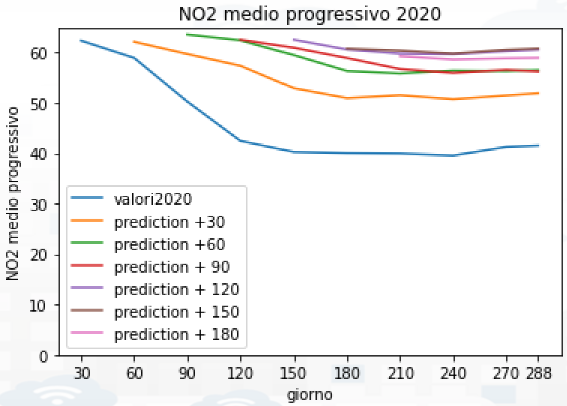
The results of the Deep LSTM Neural Networks have been compared in terms of MAPE, Mean Absolute Percentage Error results for the prediction targets of 30, 60, 90, 120, 150, and 180 days with a DNN, with the same number of layers as the LSTM architecture described in the previous section, an XGBoost and RF.

Target day	LSTM	DNN	XGBoost	RF
30	2.16	4.87	5,26	5,26
60	2.61	6.67	6,52	6,56
90	4.18	7.00	7,64	7,76
120	6.77	6.86	8,81	8,93
150	7.83	8.99	9,35	9,40
180	7.93	9.25	9,90	10

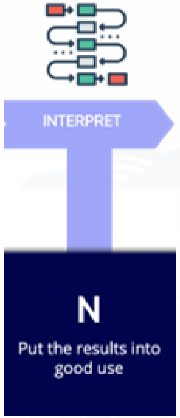
Predictions Visualization

- The results of the models for the years 2019 and 2020 are reported in the figures:





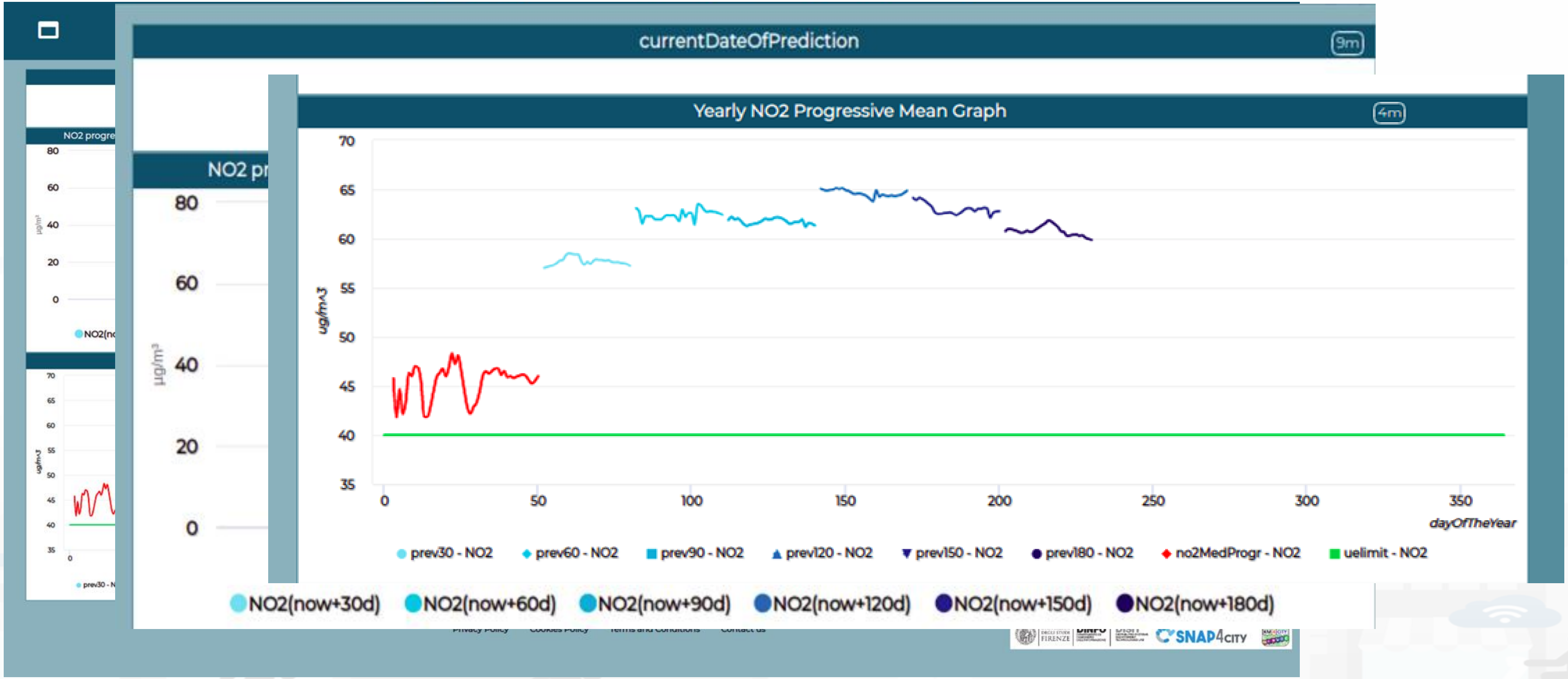
Time Series in good use- Monitoring Dashboard



- The models developed have been inserted in **an automated process** that every day generates the input for the models and makes the predictions using a service of the SNAP4City platform, the IoTApp.
- The results are saved in the infrastructure and used to generate a **dashboard** <u>to monitor the trend of the progressive mean of the NO2.</u>

Node-RED

Dashboard for Real-Time Monitoring And Prediction



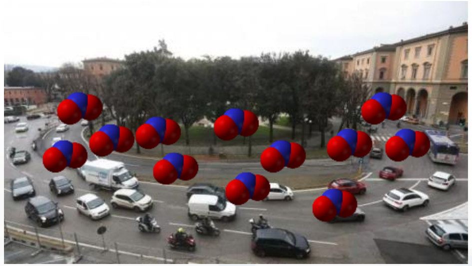
Conclusions And Future Developments

- The **dashboard** developed allows to **monitor the trend** of the progressive mean of the NO2 for the city of **Florence**, Italy.
- The results are generated by 6 **Deep LSTM Neural Networks** for the temporal targets of 30, 60, 90, 120, 150, 180 days in advance to the prediction time. These models on the test set (2019) reached results starting from a **MAPE of 1.99%** for the 30-days prediction up to a **4.18%** for the 180-days prediction.

- The presented **approach** can be applied to other air pollutants as the **PM10, CO, PM2.5** for which the UE has set limits on the yearly mean concentrations and can be **applied** to others **Smart City scenarios** assuming that the available data covers a sufficient temporal window.

What’s Next

Study the state-of-the-art AI solutions to model smart mobility **traffic flow forecasting** system from 10 to 60 minutes per 10-minute time interval on the traffic network of the **Metropolitan City of Florence**.



IEEE Access

UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB
SNAP4CITY
KM 4 CITY

Short-Term Prediction of City Vehicle Flow via Convolutional Deep Learning

City Vehicle Flow

– Traffic Flow data can be used for a number of applications:
  - Traffic Flow Analysis and reconstruction
  - What-if-analysis
  - forecasting of pollutants
– The main problem is the need of consistent data:
  - Traffic Flow sensor are not 100% reliable
  - There could be some problem in data acquisition process

 providing **PREDICTIONS** can be useful to improve quality of service

GOALS & Research Questions

- This research project has the goal to exploit a solution to compute short-term traffic flow sensors predictions up to 1 hour in advance, with a resolution of 10 minutes. The research questions at the base of this project are:

RS1) Which are the representative sensors based on a **clustering** process on the sets of traffic IoT sensor of the Metropolitan City of Florence?
RS2) Is there an **AI architecture** that achieves **best** results for the short-term prediction problem on the **case study of the Metropolitan City of Florence** compared to those used at the state of the art?
RS3) Which are the **features** actually **relevant** (historical, seasonality, weather, pollutant, etc.) in prediction computation?
RS4) How much the final architecture is **robust** for the problem of **data missing**?

Metropolitan City of Florence

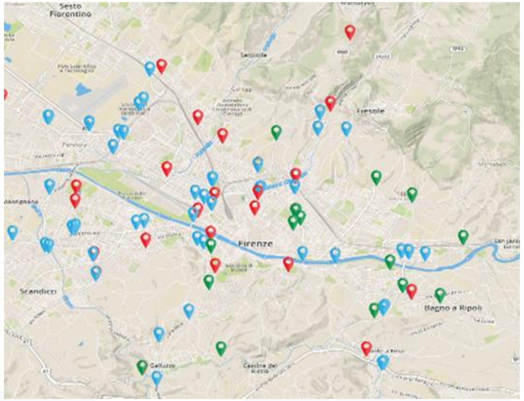
- The solution and its validation have been performed by using data collected from the sensors in the Metropolitan City of Florence
- Traffic flow in cities are tendentially **very noisy** with respect to the ones measured in high speed roads, the latter being the validation context for the majority of state of the art solutions.
- The temporal windows of data used for this research project has been from September 2019 to February 2020

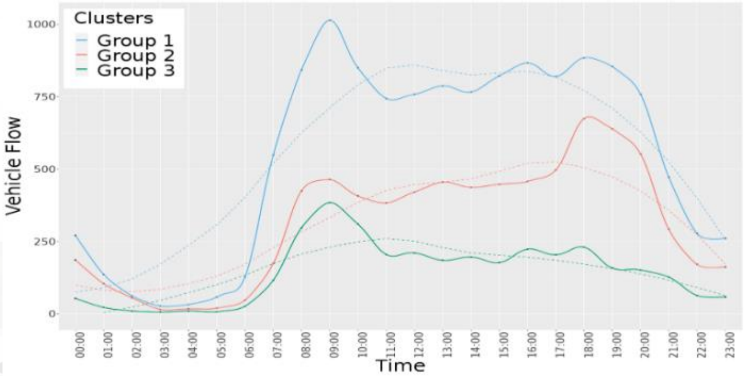
Data Clustering

- Trends of traffic flow data are strongly dependent on a number of road features:
  - road relevance (primary, secondary, etc.)
  - number of lanes, speed limits
  - presence of speed meters
  - distance from road crossing, etc
- In order to characterize the typical time trend H24 of the whole traffic flow sensors located in the city, a clustering was carried out. This approach allowed us to aggregate device sensors with the same behaviour over time.

Clustering

- The clustering has been performed on the basis of the time trend H24, considering the normalized vehicle flow measures.
- The optimal number of clusters turned out to be 3 and it has been identified by using **elbow** criteria
- **K-means** clustering method has been applied to identify clusters
  - The optimal number of clusters resulted to be equal to **3**, and it has been identified by using the **Elbow criteria**





Features

- One of the goals of this work has been conquer a general understanding above the factors that are more relevant for predicting traffic conditions in the city. Based on the related works, a set of data composed of **temporal variables, traffic-related features, weather** information, and **air pollution** has been considered.

Category	Feature	Description
Traffic Trafplus	*Vehicle Flow*	Real number of vehicles recorded every 10 minutes
	AverageSpeed	Average speed of vehicles (Km/h)
	Concentration	Number of vehicles in terms of road occupancy (%)
DateTime	*timeOfTheDay*	Time of the day {1, 144}
	dayOfTheYear	Day of the year {1, 366}
seasonality	*dayOfTheWeek*	Day of the week {1,7}
	Weekend	0 for working days, 1 else
	Year	The year of the observation
Temporal	Previous observation's difference of the previous week ()	the difference between the number of vehicles in the observation day (d) at the time slot t and the number of available bikes during the previous time slot (t-1) of the previous day (d-1)
	Subsequent observation's difference of the previous week ()	the difference between the number of vehicles in the observation day (d) at the time slot t and the number of bikes during the successive time slot (t+1) of the previous day (d-1).
	Previous week observation ()	the number of vehicles of the previous week (d-7) in the same time slot (t).

Features

- One of the goals of this work has been conquer a general understanding above the factors that are more relevant for predicting traffic conditions in the city. Based on the related works, a set of data composed of **temporal variables, traffic-related features**, **weather** information, and **air pollution** has been considered.

Category	Feature	Description
Weather	*Air Temperature*	City temperature one hour earlier than *Time* (°C)
	Humidity	City humidity one hour earlier than *Time (%)*
	Pressure	City pressure one hour earlier than *Time (millibar mb)*
	Wind Speed	City wind speed one hour earlier than *Time (KM/h)*
AirPoll	*CO*	Concentration of CO one hour earlier than *Time*
	NO2	Concentration of NO2 one hour earlier than *Time*
	O3	Concentration of O3 one hour earlier than *Time*
	PM10	Concentration of PM10 one hour earlier than *Time*
	PM2.5	Concentration of PM2.5 one hour earlier than *Time*
Weather	*Air Temperature*	City temperature one hour earlier than *Time* (°C)
	Humidity	City humidity one hour earlier than *Time (%)*

Work plan

- With a temporal target of 1h, which is the most critical short-term prediction slot ensemble learning techniques such as **Random Forest** (RF) and **Extreme Gradient Boosting Machines** (XGBOOST) are powerful techniques that must be considered for this type of problem.
- Regarding the deep learning techniques for this research project it has been proposed a new architecture **CONV-BI-LSTM** that will be compared to other solutions as **Deep Neural Network** (DNN), Deep **LSTM**, Deep **BI-LSTM** Neural Network , **Autoencoder BI-LSTM**, and an **attention-based CONV-LSTM** to assess the research question of which will be the best AI architecture for the problem of short-term prediction of vehicle flow based on this case study.

Evaluation Metrics

Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(obs_i - pred_i)^2}{n}}$$

R-Squared(R2)

- $\overline{y} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{obs_i}$
- $R^2 = 1 - \left(\frac{\sum_{i=1}^{n}(obs_i - pred_i)^2}{\sum_{i=1}^{n}(obs_i - \overline{y})^2}\right)$

Mean Absolute Scaled Error (MASE)

$$q_t = \frac{obs_t - pred_t}{\frac{1}{n-1}\sum_{i=2}^{n}|obs_i - obs_{i-1}|}$$

$$MASE = mean\,(|q_t|), \qquad t = 1, \dots, n$$

Mean Absolute Error (MAE)

$$MAE = \frac{\sum_{i=1}^{n}|obs_i - pred_i|^2}{n}$$

CONV-BI-LSTM Architecture Proposed

The defined **CONV-BI-LSTM** network is made up of 3 components:

- The first component is made up of a **Convolutional 1-dimensional layer** with 48 filters and a kernel size of 16, and a Max Pooling layer of 2x2 and stride equal to 1.
- The second component is the **BI-LSTMs** layers, in particular **6 layers** with **32 units** per layer and **dropout of 0,25**.
- The last one is made of **3 fully connected layers with number of neurons of 32-16-1**. The last one has a sigmoid activation to produce the prediction.

The used optimizer is **Adam Optimizer** with learning rate between **0.005 and 0.008**. **MSE** was selected as the loss function to be monitored during **optimization**. The **batch size** has been set to **512** and the number of epochs was set to a maximum value of 1000, because the training strategy used the **Early Stopping** method with patience parameter set to 100 to determine the optimum epoch number minimizing the MSE of the validation set, restoring the weights of the best model at the end of the learning process.

Results

TABLE VI – THE MAPE ESTIMATED FOR 64 COMBINATIONS OF FEATURES FOR ALL THE IDENTIFIED TECHNIQUES AS THE MEDIAN VALUE ON THE SENSORS IN THE 3 CLUSTERS DESCRIBED ABOVE. THE ORDER IS BASED ON THE COMBINATION OF FEATURES. IN BOLD, BEST RESULTS/CONFIGURATIONS.
IN BOLD WITH CITATION: RESULTS OBTAINED TAKING INTO ACCOUNT SOLUTIONS FROM THE STATE OF THE ART.

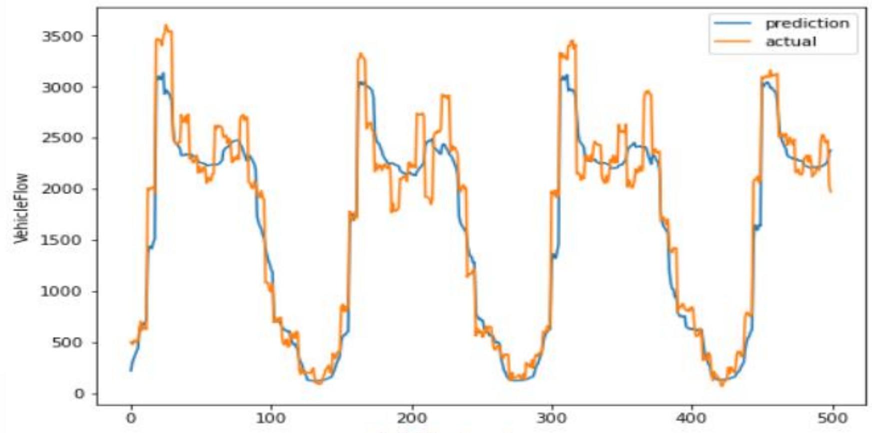
ID	Features adopted in the model						Median value of MAPE for prediction results by technique								min
	Date time	Traf plus	Tempo ral	Seasona lity	Airpoll	weather	RF	XGBOOS T	DNN	LSTM	BI-LSTM	Autoencode rBI-LSTM	Attention CONV-LSTM	CONV-BI-LSTM	
C1	Y	Y	Y	Y	Y	Y	29.342	34.552	42.754	49.407	34.865	34,708	37,059	31.365	29.342
C2	Y	Y	Y	Y	Y	N	29.682	35.545	43.400	49.832	35.870	35,707	39,506	35.613	29.682
C3	Y	Y	Y	Y	N	Y	28.782	34.441	35.465	36.824	31.555	32,998	33,179	30.894	28.782
C4	Y	Y	Y	Y	N	N	30.935	35.373	38.942	35.383	30.564	32,969	35,713	32.485	30.564
C5	Y	Y	Y	N	Y	Y	29.776	34.469	33.425	42.301	39.865	37,167	35,161	36.897	29.776
C6	**Y**	**Y**	**Y**	**N**	**Y**	**N**	**29.598**	**35.547**	**33.865**	**36.792**	**35.097**	35,322	29,923	**25.981**	**25.981**
C7	Y	Y	Y	N	N	Y	29.421	33.711	31.377	34.736	40.510	37,110	30,741	30.106	29.421
C8	Y	Y	Y	N	N	N	31.245	34.414	32.026	37.823	40.662	37,538	31,263	30.500	30.500
C9	Y	Y	N	Y	Y	Y	29.626	36.919	42.187	**37.068 [38]**	34.297	35,608	36,651	**31.115**	29.626
C10	Y	Y	N	Y	Y	N	29.964	35.802	47.201	41.334	34.743	35,272	40,658	34.116	29.964
C11	Y	Y	N	Y	N	Y	29.785	35.976	45.451	44.756	41.620	38,798	37,345	29.240	29.240
C12	Y	Y	N	Y	N	N	31.262	35.792	36.040	37.228	32.727	34,259	32,701	29.363	29.363
C13	Y	Y	N	N	Y	Y	29.431	35.935	34.448	35.829	34.619	35,277	32,287	30.126	29.431
C14	Y	Y	N	N	Y	N	29.764	36.374	36.203	43.510	35.744	36,059	33,015	29.827	29.764
C15	Y	Y	N	N	N	Y	29.972	35.423	31.526	46.201	37.209	36,316	32,919	34.313	29.972
C16	Y	Y	N	N	N	N	**30.960 [14]**	34.235	30.338	**37.068 [23]**	**38.082 [39]**	**34,235 [45]**	**29,455[46]**	**28.573**	28.573
C17	Y	N	Y	Y	Y	Y	29.281	34.503	72.909	64.557	48.685	41,594	51,026	29.144	29.144
C18	Y	N	Y	Y	Y	N	30.184	35.350	59.458	68.127	46.874	41,112	44,810	30.163	30.163
C19	Y	N	Y	Y	N	Y	28.711	34.316	45.679	46.211	33.404	33,86	37,125	28.571	28.571
C20	Y	N	Y	Y	N	N	31.211	34.784	51.603	45.188	48.643	41,713	40,862	30.122	30.122
C21	Y	N	Y	N	Y	Y	30.689	35.774	36.428	48.608	40.092	37,933	34,801	33.175	30.689
C22	Y	N	Y	N	Y	N	30.505	36.165	37.337	61.168	34.420	35,292	34,385	31.434	30.505
C23	Y	N	Y	N	N	Y	30.036	34.779	37.583	64.341	51.063	42,921	33,455	29.328	29.328
C24	Y	N	Y	N	N	N	32.629	34.312	36.849	53.854	41.912	38,112	33,257	29.665	29.665
C25	Y	N	N	Y	Y	Y	28.766	35.906	71.829	65.565	54.403	45,154	52,023	32.218	28.766
C26	Y	N	N	Y	Y	N	30.008	37.317	67.870	49.366	46.880	42,098	53,256	38.642	30.008
C27	Y	N	N	Y	N	Y	28.986	35.218	57.938	50.333	59.419	47,318	43,298	28.658	28.658
C28	**Y**	**N**	**N**	**Y**	**N**	**N**	**31.068**	**35.878**	**66.634**	**50.957**	**55.096**	45,487	47,097	**27.561**	**27.561**
C29	Y	N	N	N	Y	Y	29.301	37.532	38.325	40.677	50.303	43,917	35,554	32.784	29.301
C30	Y	N	N	N	Y	N	29.323	37.284	37.149	48.801	55.064	46,174	34,721	32.294	29.323
C31	Y	N	N	N	N	Y	29.964	36.331	34.638	56.157	45.016	40,673	35,293	35.949	29.964
C32	**Y**	**N**	**N**	**N**	**N**	**N**	**29.281**	**34.574**	**33.028**	**57.961**	**44.977**	39,775	29,320	**25.612**	**25.612**
C33	N	Y	Y	Y	Y	Y	61.579	71.245	77.572	82.634	49.253	60,249	62,308	47.044	47.044
C34	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Results

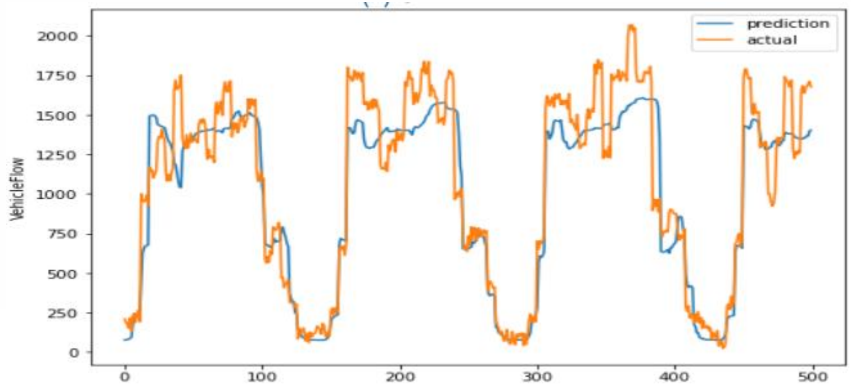
- The proposed architecture achieved promising results based on the evaluation metrics introduced.

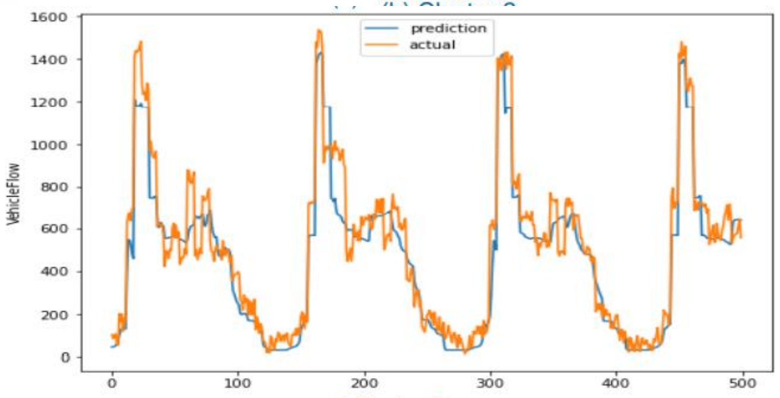
Representative sensor	MAE	RMSE	R2	MASE
Cluster-1	161,42	221,84	0.95	0.51
Cluster-2	138,98	182,48	0.90	0.60
Cluster-3	81,86	124,82	0.89	0.57

Predictions On Representative Sensors



(a) Cluster 1





l Cluster 3

FEATURE CATEGORY IMPORTANCE ANALYSIS

- It has been performed a feature importance analysis using the CONV-BI-LSTM model on the representative sensor of Cluster-1.
- The analysis calculated the MAPEs using all the features except the specific considered category excluding recursively each single feature category
- The DMAPE is defined as the difference of MAPE with respect to the minimum MAPE registered for the CONV-BI-LSTM such as:

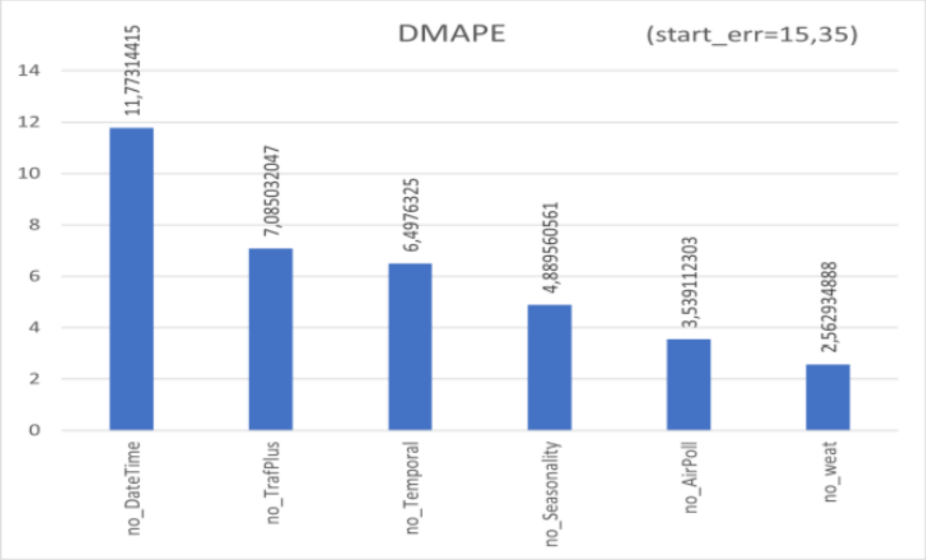
$$DMAPE_i = MAPE_{all-cat_i} - minMAPE$$

Where: i = 1, …, number of categories-1 (all, except the traffic, for a total of 6).

Categories with a higher DMAPE are the most relevant ones, since they do not cause larger differences / errors.

FEATURE CATEGORY IMPORTANCE ANALYSIS

- The feature category with the highest DMAPE is the **DateTime** followed by the **Trafplus**, and the **Temporal feature** category. Additional information on data seasonality for short-term prediction has been ranked 4th , ahead of Air Pollution feature category which in turn beats also Weather features



IMPACT OF DATA MISSING ON PRECISION

- **Data missing is an inevitable problem** when dealing with real world IoT sensor networks. Traffic sensors may suffer of problems such as detector malfunction and communication failure.
- The presence of missing data samples in making predictions (execution of the predictive model)may impact on the precision, up to make **impossible to produce the prediction**
- The approach of data imputation can be a valid option to produce surrogate data.
- In this case it has been used an **Hot-Deck** imputation.

data missing robustness

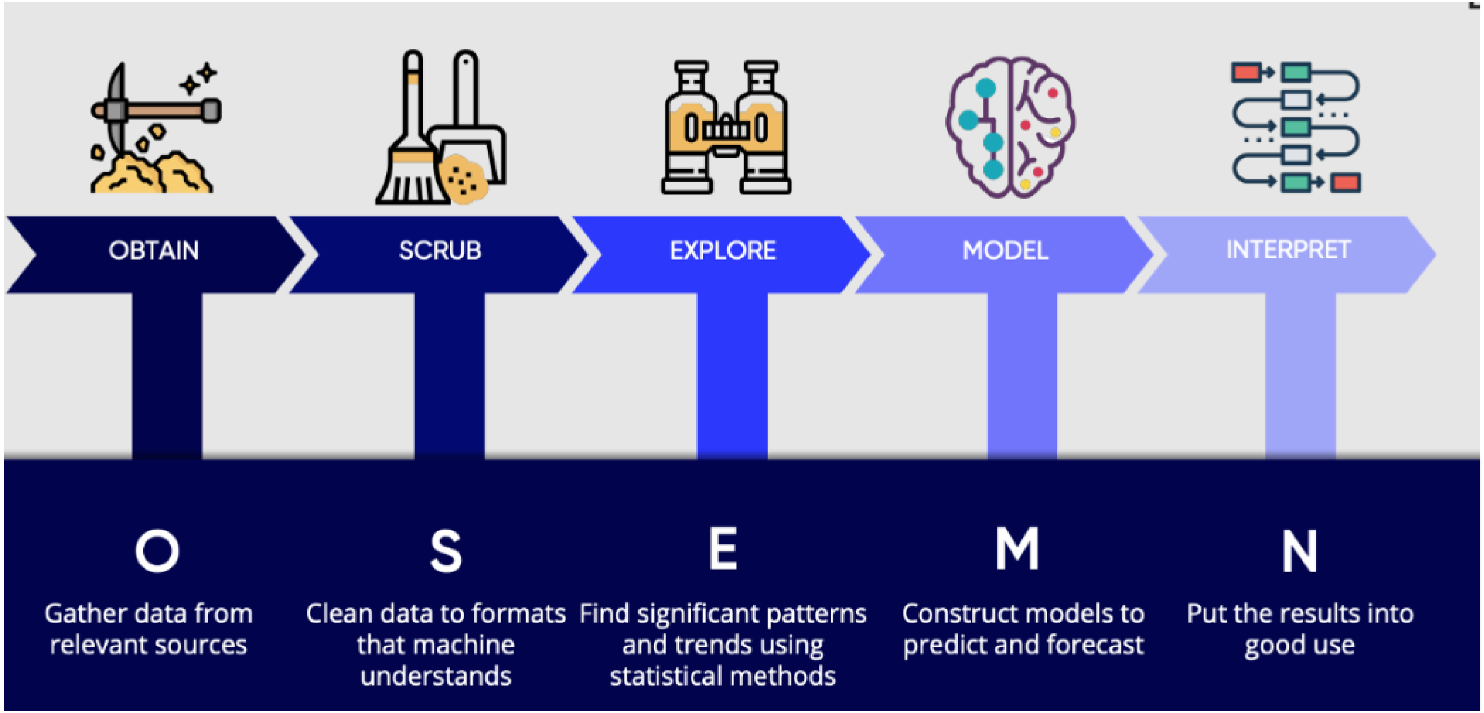
- The robustness has been assessed on the test dataset from 10/02/2020 to 16/02/2020 randomly setting to missing the Vehicle Flow of a percentage of the total dataset based on the missing rates chosen (10%, 25%, 50%, 75%) and then imputing the missing data.
- The imputation strategy proposed to handle missing data reports valid results for the missing rates of 10%, 25%, 50%, 75% on all the representative sensors of the three clusters.

TABLE VIII - DATA MISSING ANALYSIS BASED ON DIFFERENT MISSING RATES ON THE CLUSTERS REPRESENTATIVE SENSORS OF TABLE VII.

Representative sensor	Missing Rate	MAE	MAPE	RMSE	R2
cluster 1 METRO775	0%	161.42	15.35	221.84	0.95
	10%	173.19	16.12	241.86	0.94
	25%	177.36	17.17	258.88	0.93
	50%	176.98	16.77	258.26	0.93
	75%	173.92	16.67	248.51	0.93
cluster 2 METRO707	0%	138.98	23.86	182.48	0.90
	10%	147.49	25.36	194.64	0.88
	25%	146.77	24.90	193.56	0.88
	50%	145.72	24.52	193.34	0.88
	75%	146.10	24.58	193.46	0.88
cluster 3 METRO714	0%	81.86	25.73	117.37	0.89
	10%	83.73	27.99	119.32	0.87
	25%	83.01	27.15	119.11	0.87
	50%	85.00	28.92	122.33	0.87
	75%	82.18	26.89	118.42	0.88

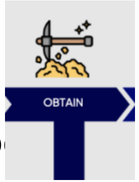
Conclusions

- Accessing precise traffic flow data is mandatory to guarantee **high level of services** such as: traffic flow reconstruction, which in turn is used to perform what-if analysis, conditioned routing, etc. They have to be **reliable and precise for possible rescue teams** and fire brigades.
- It has been conducted a **clustering** process to determine the 3 main representative sensors of the road network of the Metropolitan City of Florence
- The **proposed architecture** achieved promising preliminary results for the short-term 1h prediction of city vehicle flow on all the representative sensors
- The most important feature categories are the **DateTime** followed by the **Trafplus**, and the **Temporal** feature category. The weather data are not so relevant despite what is reported in the state-of-the-art.
- The solution using a **Hot Deck data imputation strategy is robust on eventual data missing** events.

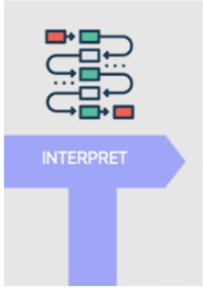


Enrico Collini

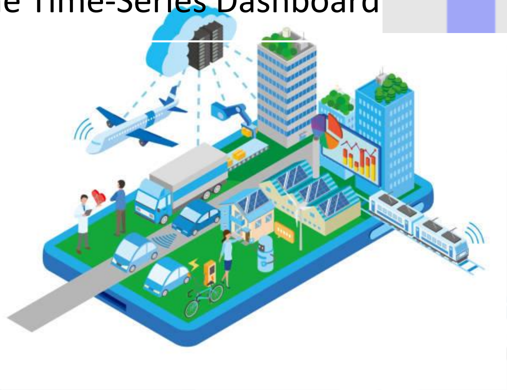
Interpret Data Analytics using Snap4City



- Retrive Data from Snap4City Op
  - login
  - documentation
- Data Analytic Node-Red Python Container
- Private Device Data Retrieval
- Snap4City Real-Time Time-Series Dashboard



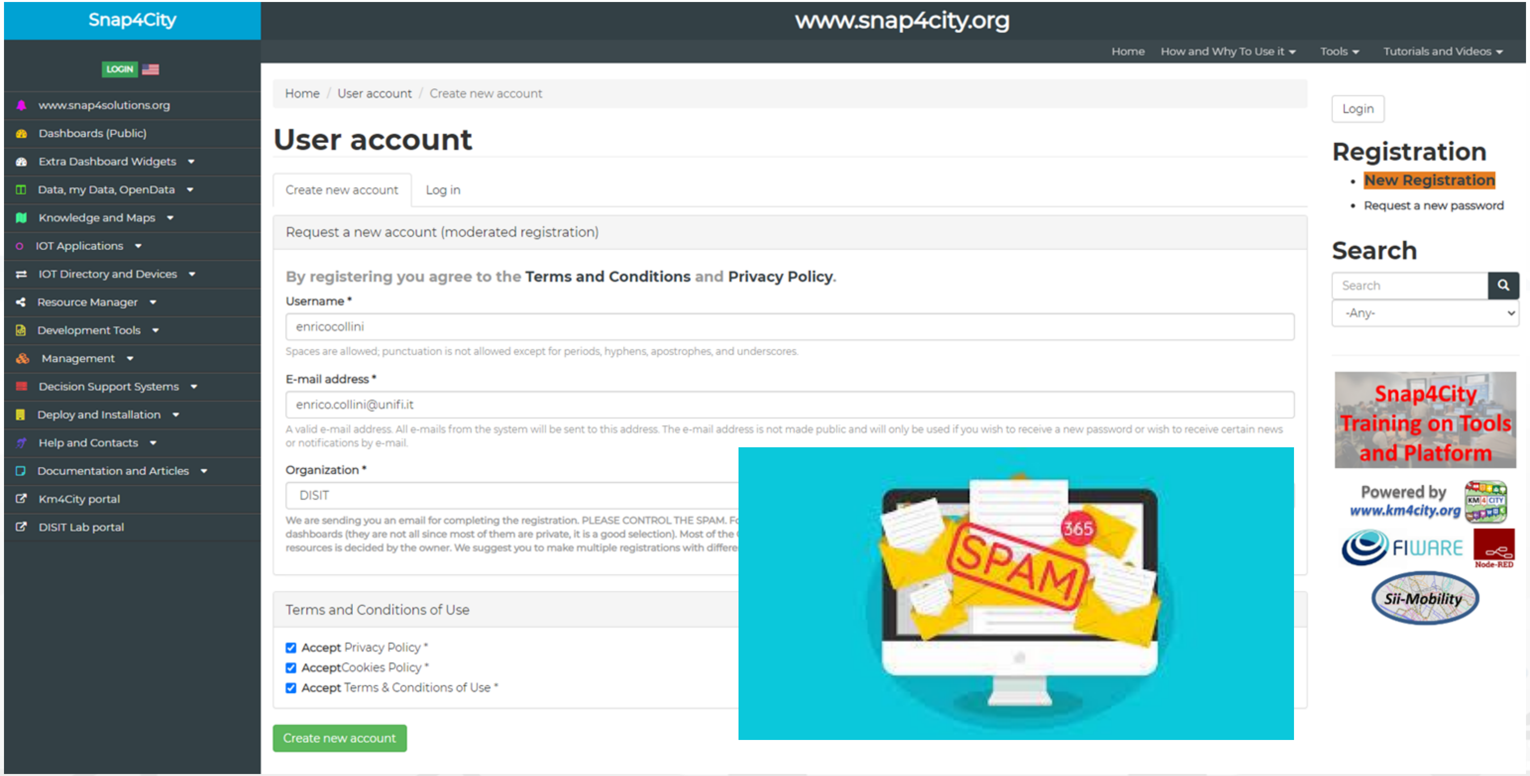
python™



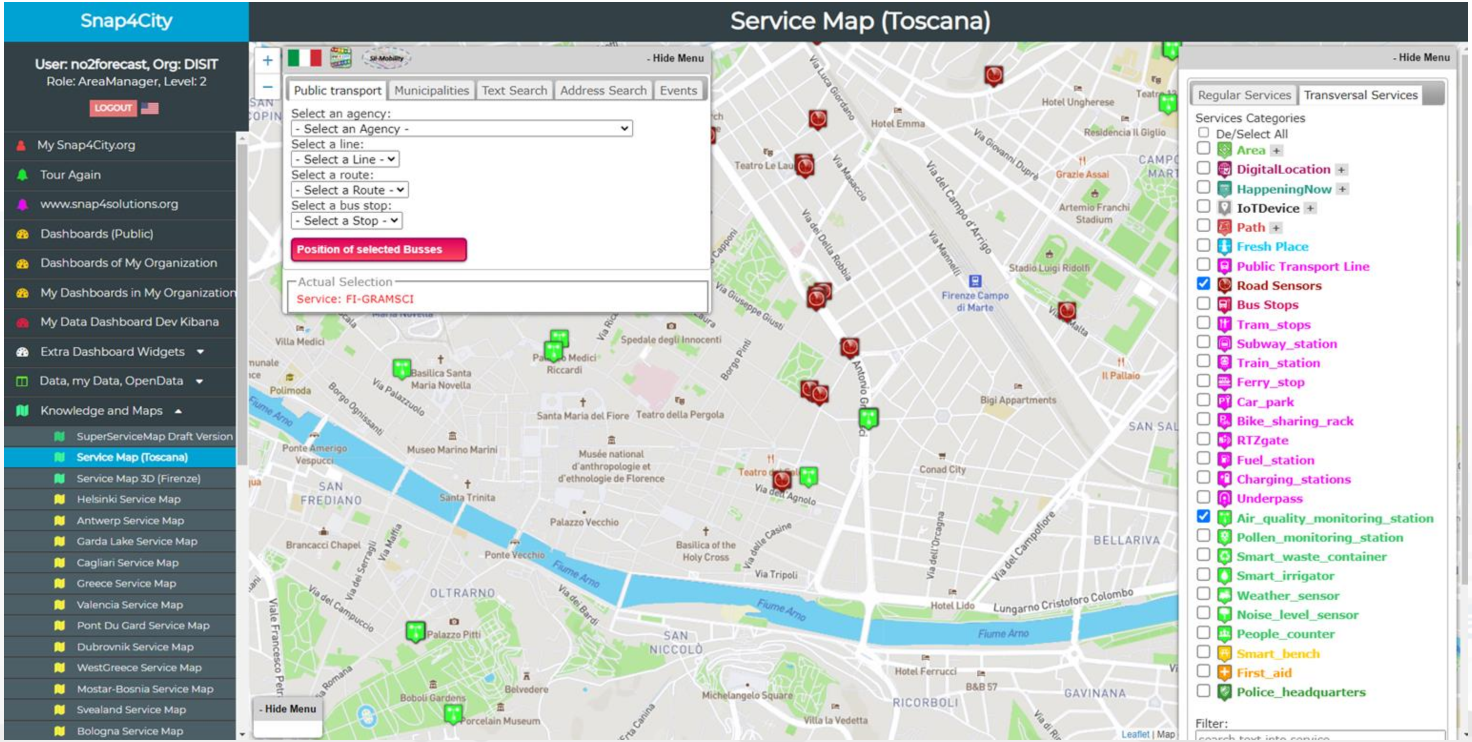
Node-RED

Create a Snap4city account

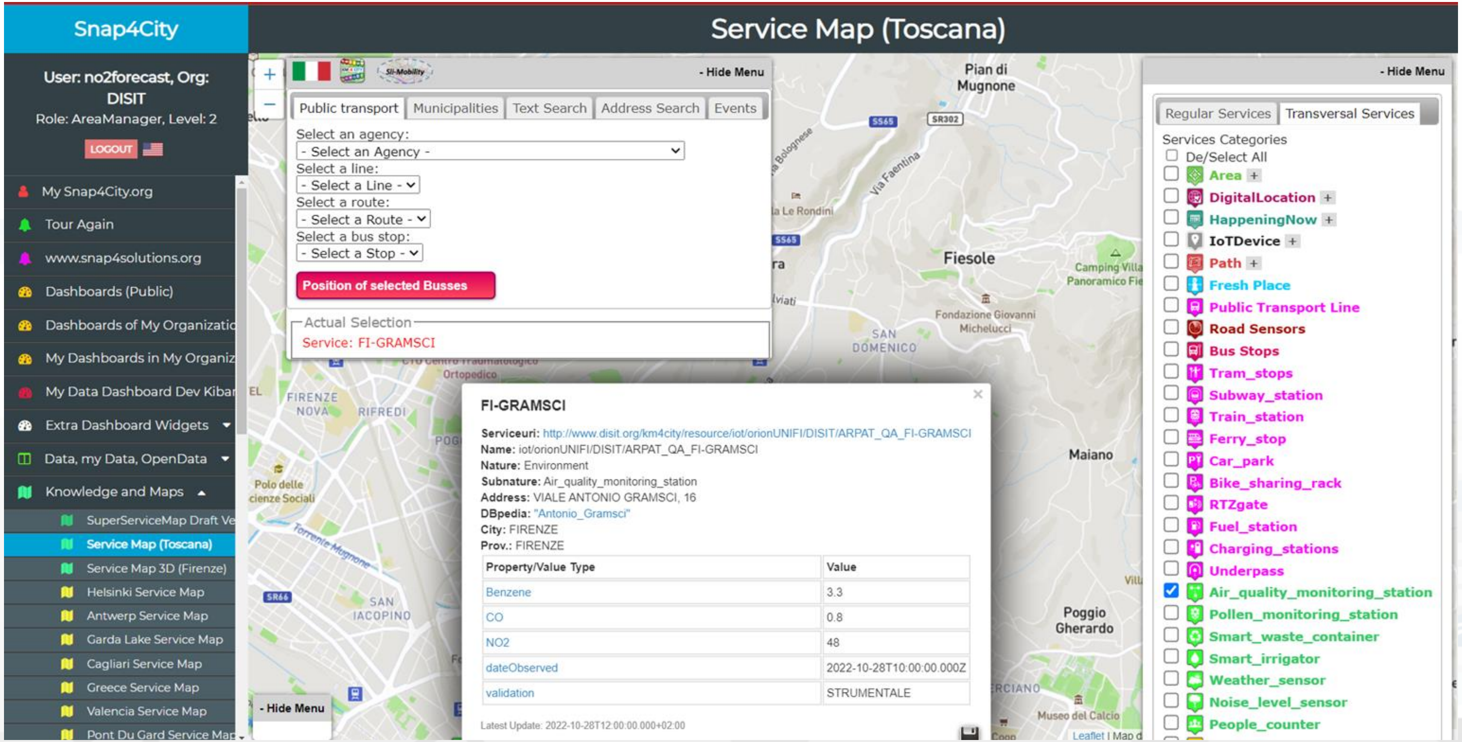
Create a Snap4city account



Choose a Sensor in ServiceMap



Choose a Sensor in ServiceMap



Choose a Sensor in ServiceMap

FI-GRAMSCI

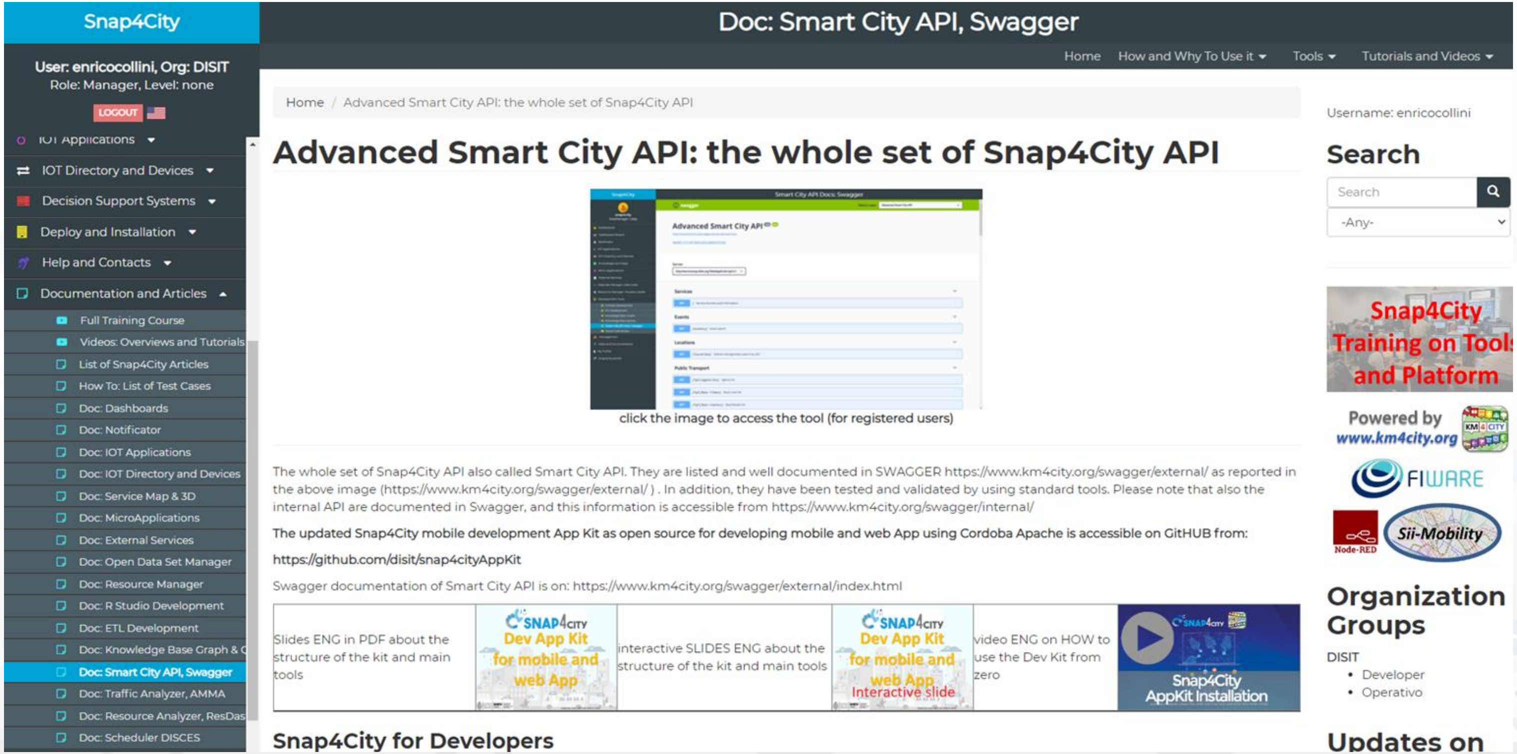
Serviceuri: http://www.disit.org/km4city/resource/iot/orionUNIFI/DISIT/ARPAT_QA_FI-GRAMSCI
Name: iot/orionUNIFI/DISIT/ARPAT_QA_FI-GRAMSCI
Nature: Environment
Subnature: Air_quality_monitoring_station
Address: VIALE ANTONIO GRAMSCI, 16
DBpedia: "Antonio_Gramsci"
City: FIRENZE
Prov.: FIRENZE

Property/Value Type	Value
Benzene	3.3
CO	0.8
NO2	48
dateObserved	2022-10-28T10:00:00.000Z
validation	STRUMENTALE

Latest Update: 2022-10-28T12:00:00.000+02:00

http://www.disit.org/km4city/resource/iot/orionUNIFI/DISIT/ARPAT_QA_FI-GRAMSCI

Access the API documentation



Acces the API documentation

Services

GET / Service discovery and information

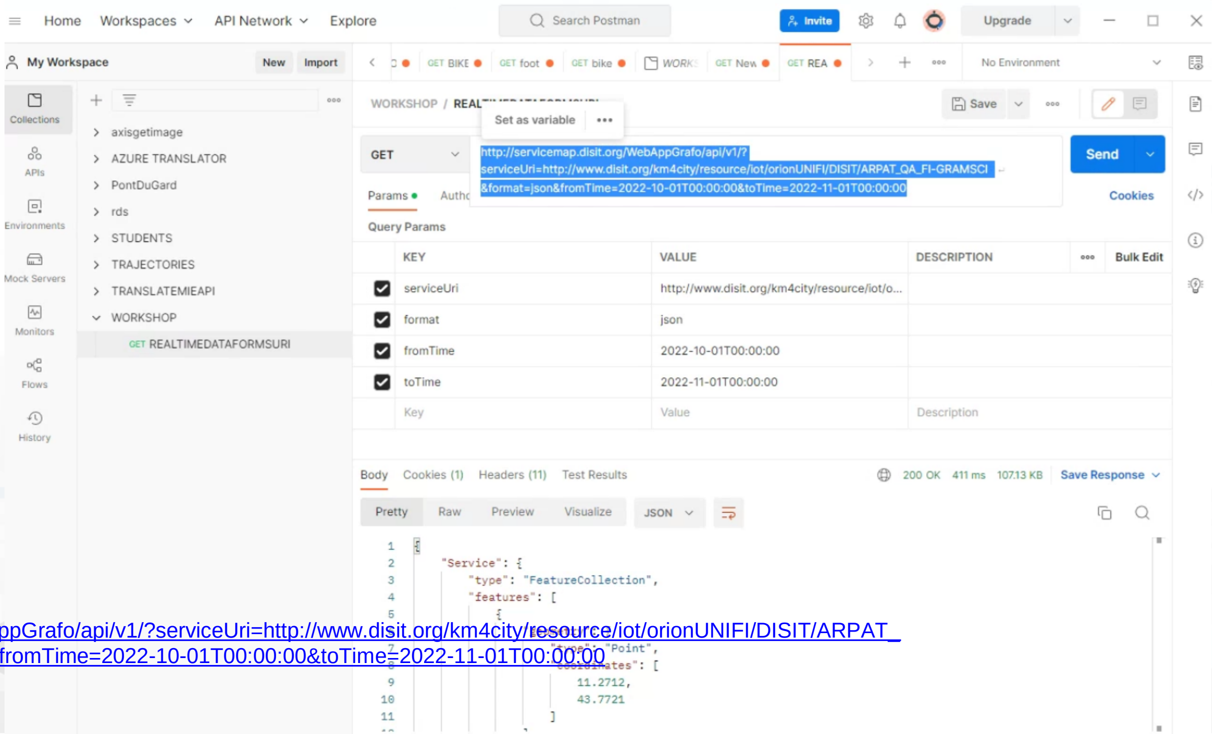
- **Service search near GPS position** - It allows to retrieve the set of services that are near a given GPS position. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field. It can also be used to find services that have a WKT spatial description that contains a specific GPS position.
- **Service search near a service** - It allows to retrieve the set of services that are near a given service identified by its *serviceUri*. The services can be filtered as belonging to specific categories (e.g. Accomodation, Hotel, Restaurant, etc.), or having specific words in any textual field. It can also be used to find services that have a WKT spatial description that contains a specific GPS position.
- **Service search within a GPS area** - It allows to retrieve the set of services that are inside a rectangular area. The services can be filtered as belonging to specific categories (e.g. Accomodation, Hotel, Restaurant, etc.), or having specific words in any textual field.
- **Service search within a WKT described area** - It allows to retrieve the set of services that are inside a geographic region described using the Well Known Text (WKT) format. The services can be filtered as belonging to specific categories (e.g. Accomodation, Hotel, Restaurant, etc.), or having specific words in any textual field.
- **Service search within a stored WKT described area** - It allows to retrieve the set of services that are inside a geographic region described using the Well Known Text (WKT) format, by referring to the WKT with an identifier provided when the WKT is stored. The services can be filtered as belonging to specific categories (e.g. Accomodation, Hotel, Restaurant, etc.), or having specific words in any textual field. The list of available geometries can be retrived from the Service Map in the *Search Area* selection box (with *Search Range specific area*). New geometries can be provided using the http://www.km4city.org/wkt web service which can store a WKT from a shp file or providing directly the WKT string.
- **Service search by municipality** - It allows to retrieve the set of services that are in a specific municipality. The services can be filtered as belonging to specific categories (e.g. Accomodation, Hotel, Restaurant, etc.), or having specific words in any textual field.
- **Service search by query id** - It allows to retrieve the set of services associated with a query stored using the Service Map user interface.
- **Full text search** - It allows to retrieve the geolocated entities (not only services) that match with a list of keywords. The results can be possibly filtered to be within a specified distance from a GPS position, or within a rectangular area or inside a WKT geolocated area.
- **Service info** - It allows to retrieve information about a service using its *serviceUri*, as an HTML (*format* query parameter set to *html*) or a machine readable JSON document (*format* query parameter set to *json*).

Parameters

Try it out

Test the API

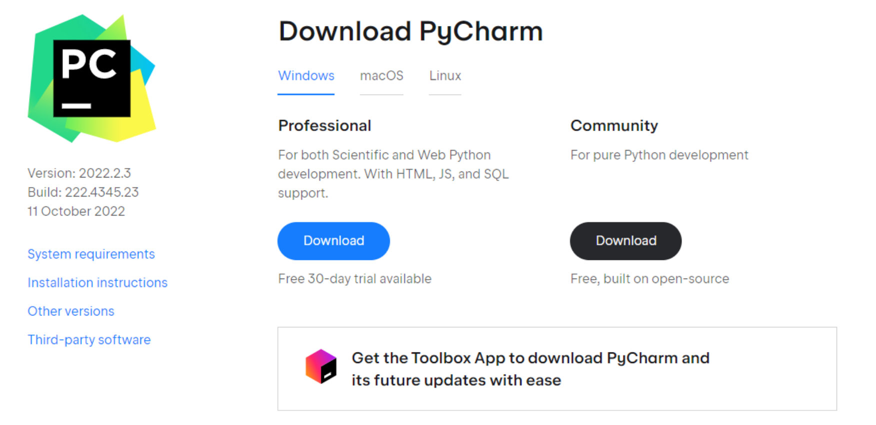
POSTMAN

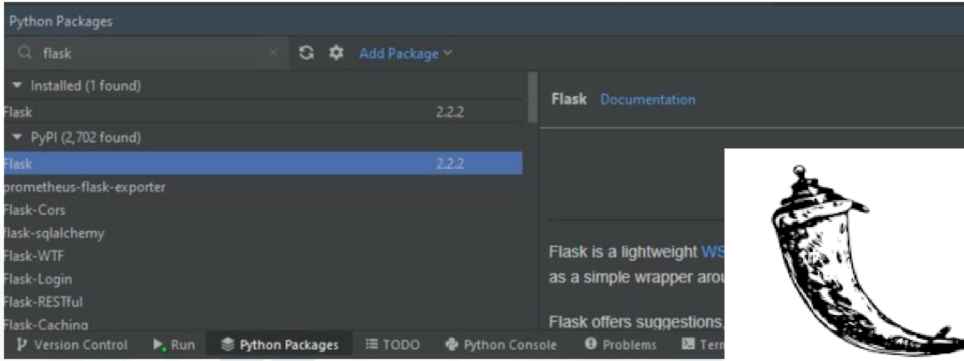


http://servicemap.disit.org/WebAppGrafo/api/v1/?serviceUri=http://www.disit.org/km4city/resource/iot/orionUNIFI/DISIT/ARPAT_QA_FI-GRAMSCI&format=json&fromTime=2022-10-01T00:00:00&toTime=2022-11-01T00:00:00

Flask Python local Development

- install python in your environment
  - Windows: https://www.python.org/downloads/
  - Linux `sudo apt-get install python`
- Suggested IDe





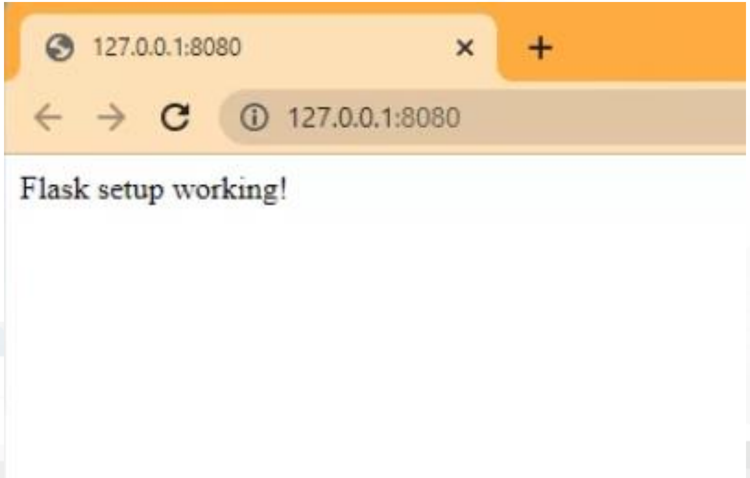

```
pip install Flask
```

Flask Python local Development

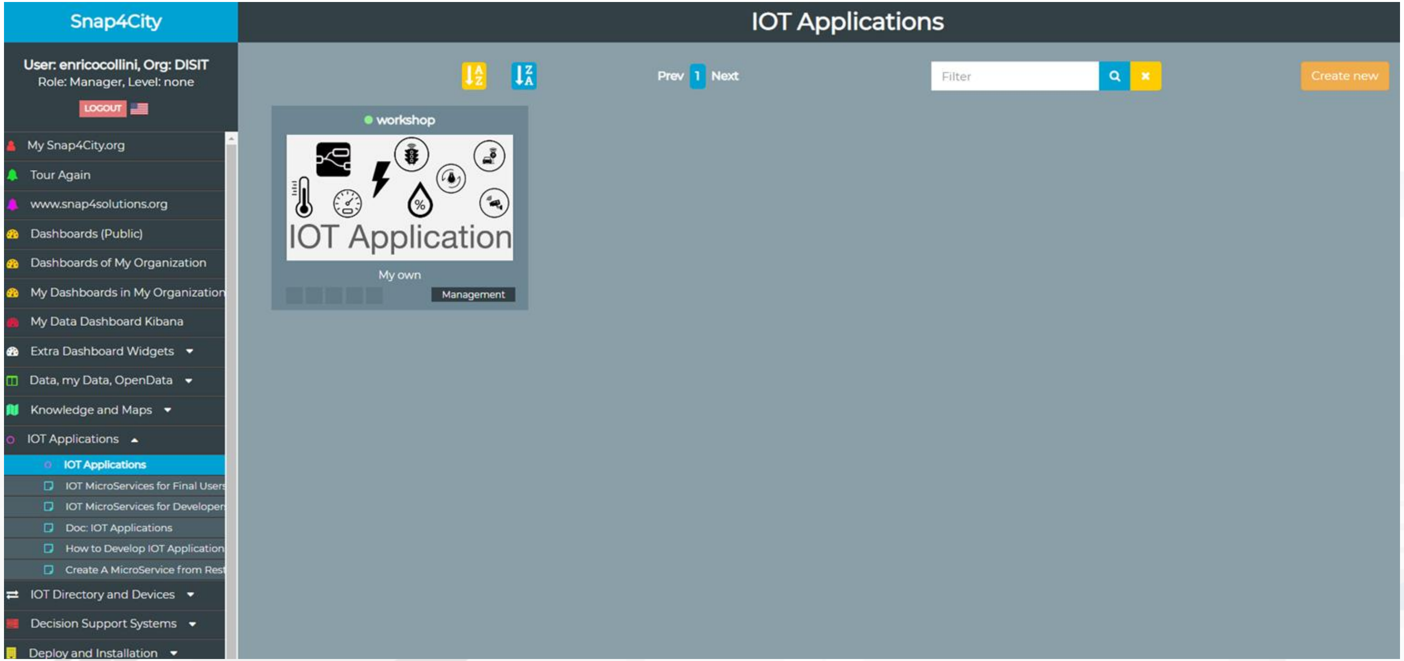
testflask - Blocco note di Windows

File Modifica Formato Visualizza ?

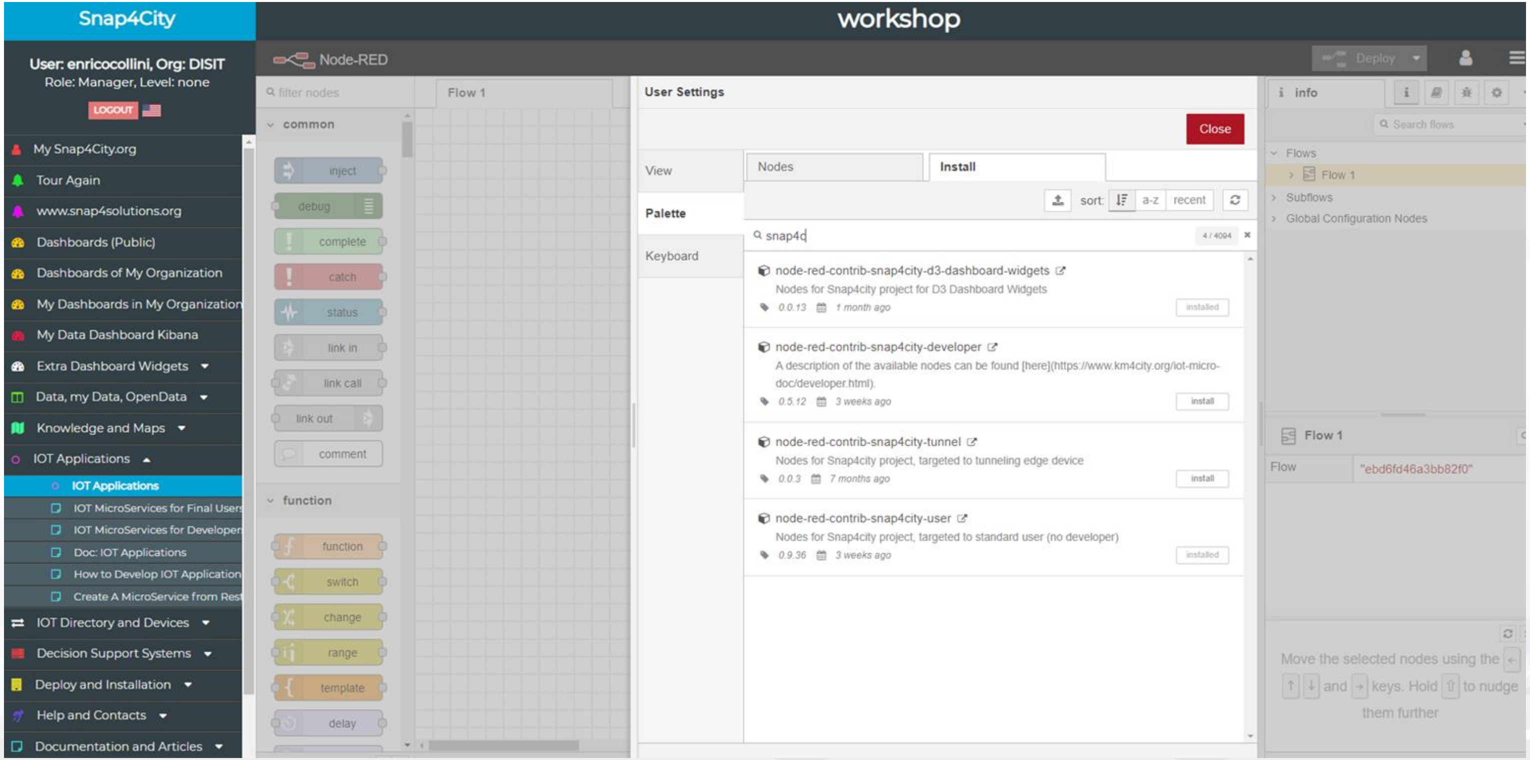
```
import flask
app = flask.Flask(__name__)
#python decorator that adds functionality to the function below
@app.route("/")
def index():
    return "Flask setup working!"
app.run(host='0.0.0.0', port=8080)
```



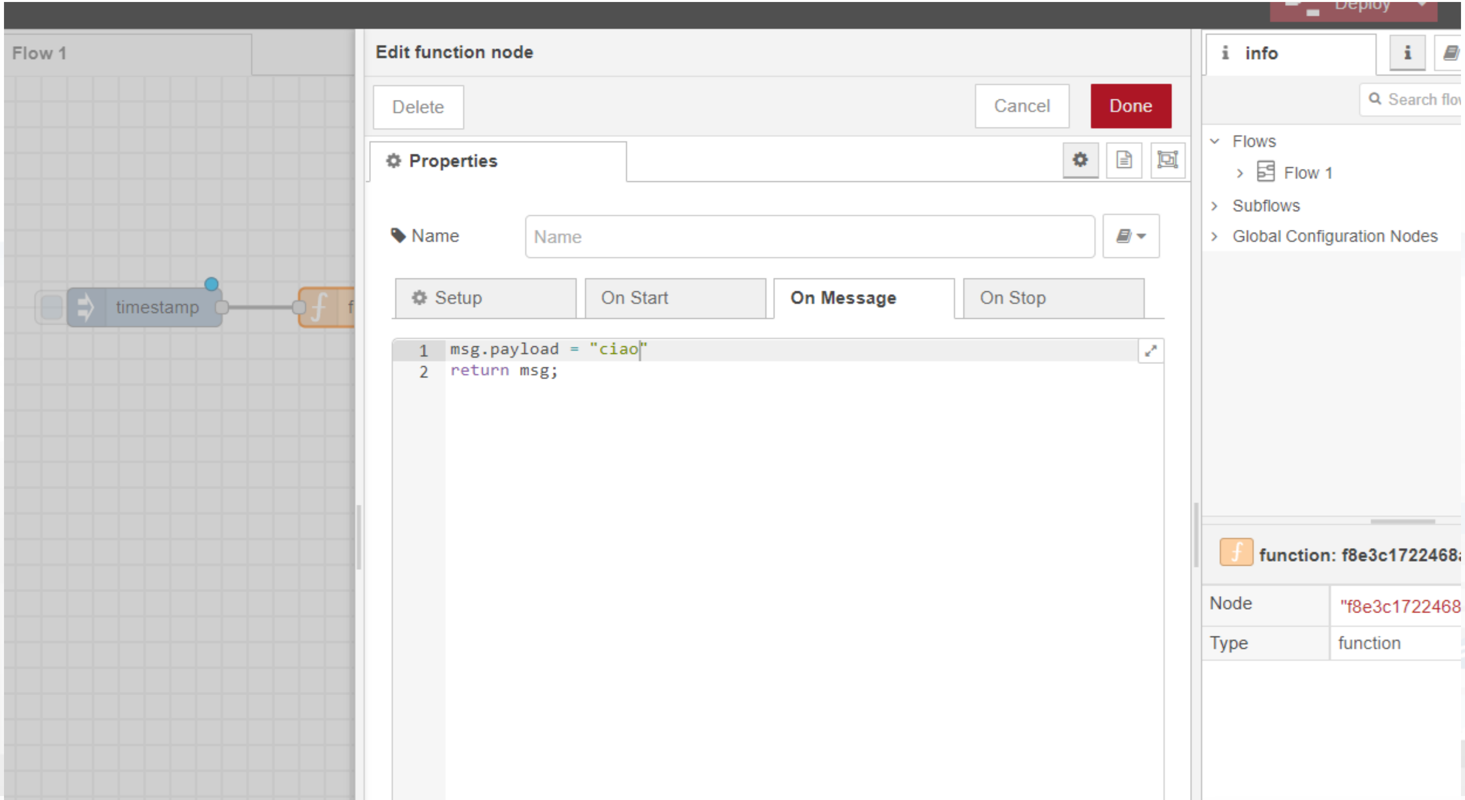
Data Analytic Node-Red Python Container



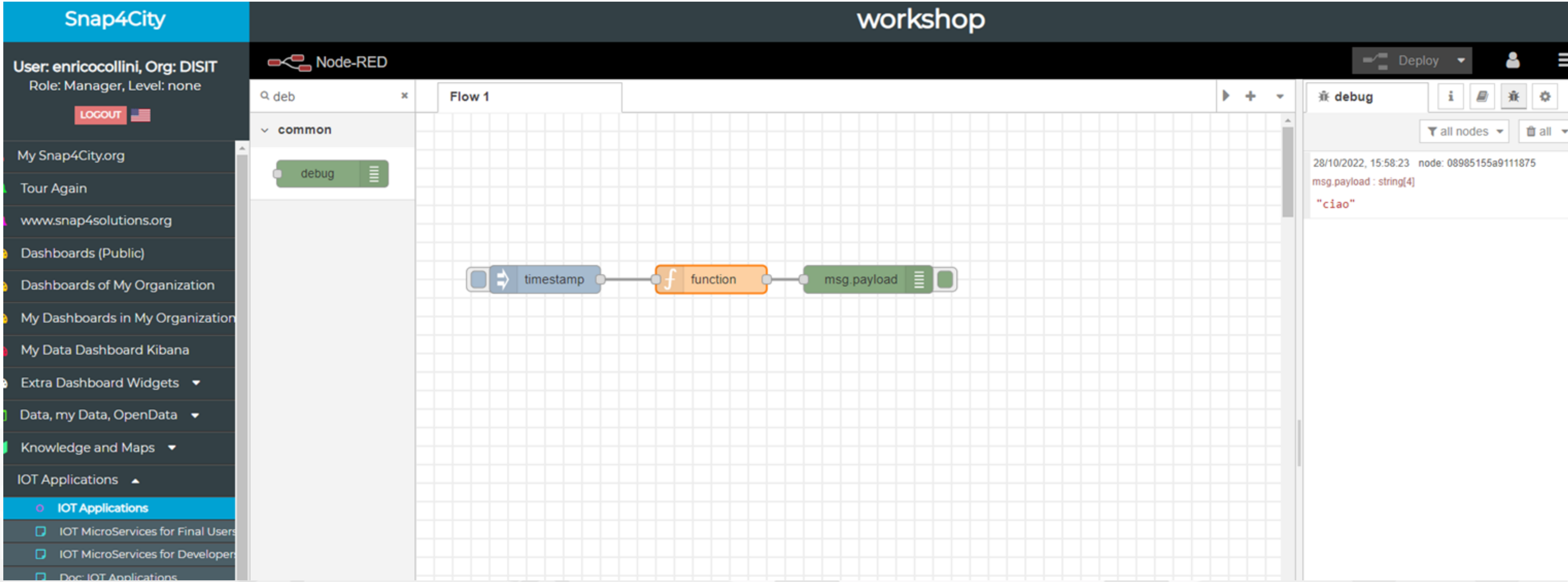
Data Analytic Node-Red Python Container



Data Analytic Node-Red Python Container



Data Analytic Node-Red Python Container



Setup Completed

Python API data Request

```
pip install requests
```

```python
import requests

suri= "http://www.disit.org/km4city/resource/iot/orionUNIFI/DISIT/ARPAT_QA_FI-GRAMSCI"
fromTime = "2022-10-01T00:00:00"
fromTime = "2022-11-01T00:00:00"
api_request = "http://servicemap.disit.org/WebAppGrafo/api/v1/?serviceUri=" + suri + \
              "&format=json&fromTime=" + fromTime + "&toTime=" + fromTime
response = requests.get(api_request)
payload = response.json()
print(payload)
```

```
etup\venv\Scripts\python.exe" "C:/Users/collini/PycharmProjects/flask setup/main.py"
'features': [{'geometry': {'type': 'Point', 'coordinates': [11.2712, 43.7721]}, 'type': 'Feature',
```

main.py

```python
import flask
import requests
app = flask.Flask(__name__)
def getDatafromSensorURI(suriSpecifico, fromTime, toTime):
    geturl = "http://servicemap.disit.org/WebAppGrafo/api/v1/?serviceUri=" + suriSpecifico + \
             "&format=json&fromTime=" + fromTime + "&toTime=" + toTime
    print(geturl)
    response = requests.get(geturl)
    payload = response.json()
    return payload
@app.route('/scriptBello', methods=['GET', 'POST'])
def scriptBello(): # GET PARAMS IN INPUT (day_date,sensor_uri)
    try:
        if flask.request.method == 'GET':
            start_date = str(flask.request.values.get('start_date')) #2022-10-01T00:00:00
            end_date = str(flask.request.values.get('end_date'))
            sensor_uri = str(flask.request.values.get('sensor_uri'))
            print("start_date:", start_date)
            print("end_date:", end_date)
            print("sensor_uri:", sensor_uri)
        else:
            yourarg = "Nothing"
            print("Nothing: ")
    except Exception as e:
        print("Error: " + str(e))
        message = "Error: " + str(e);
        return message
    try:
        res = getDatafromSensorURI(sensor_uri, start_date, end_date)
        return res
    except Exception as e:
        print("Error: " + str(e))
        message = "Error: " + str(e);
        return message
if __name__ == "__main__":
    app.run(host='0.0.0.0', port=8080)
```

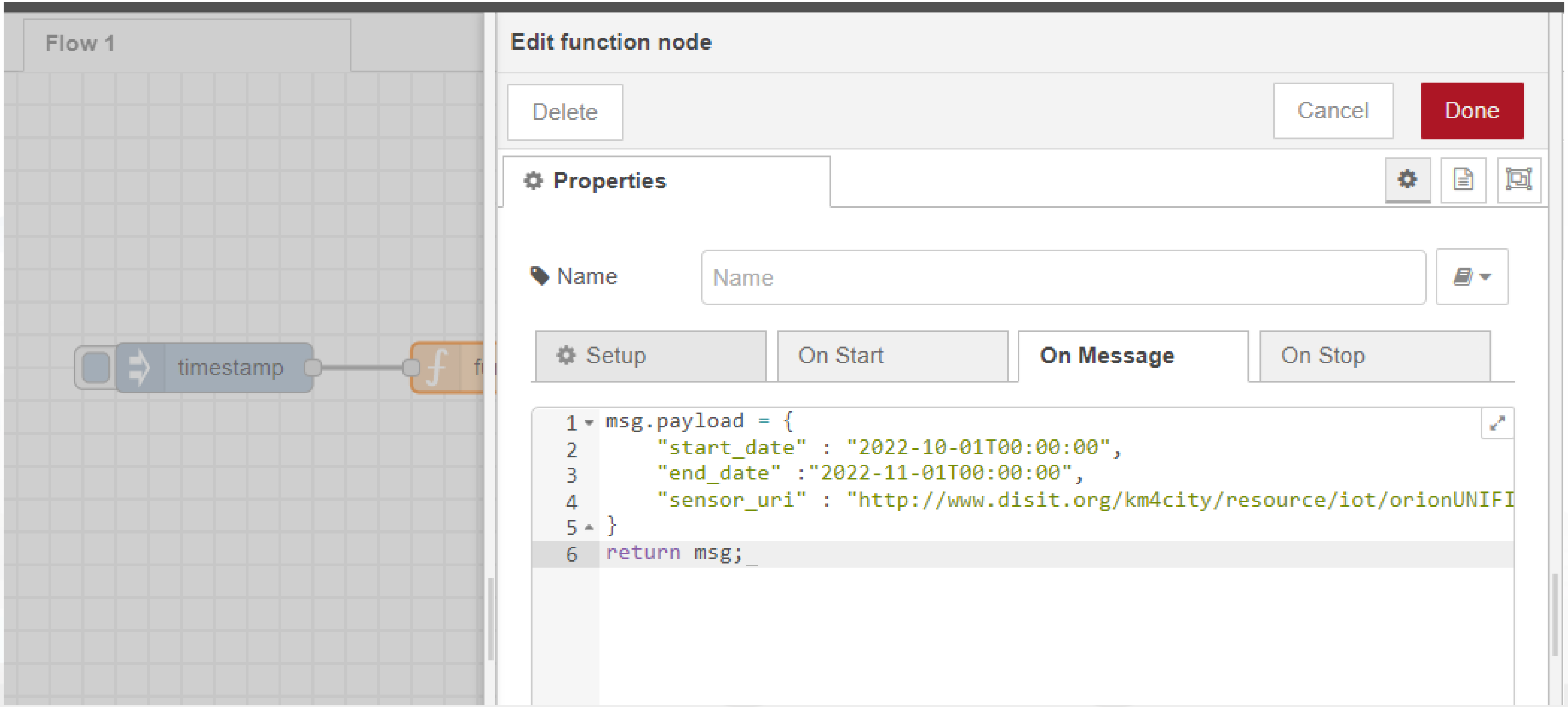
SNAP4CITY

/scriptBello

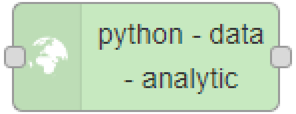
to get data of a device identified by its sernsor_uri from start_date to end_date

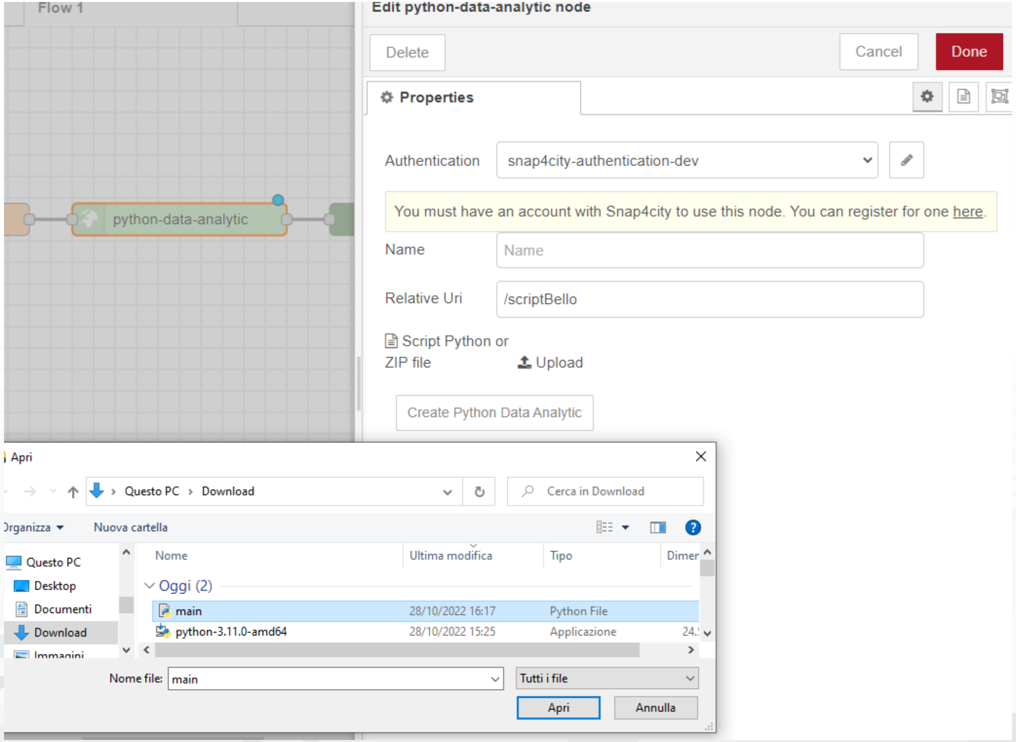
structure required for the python iot app NodeRed block

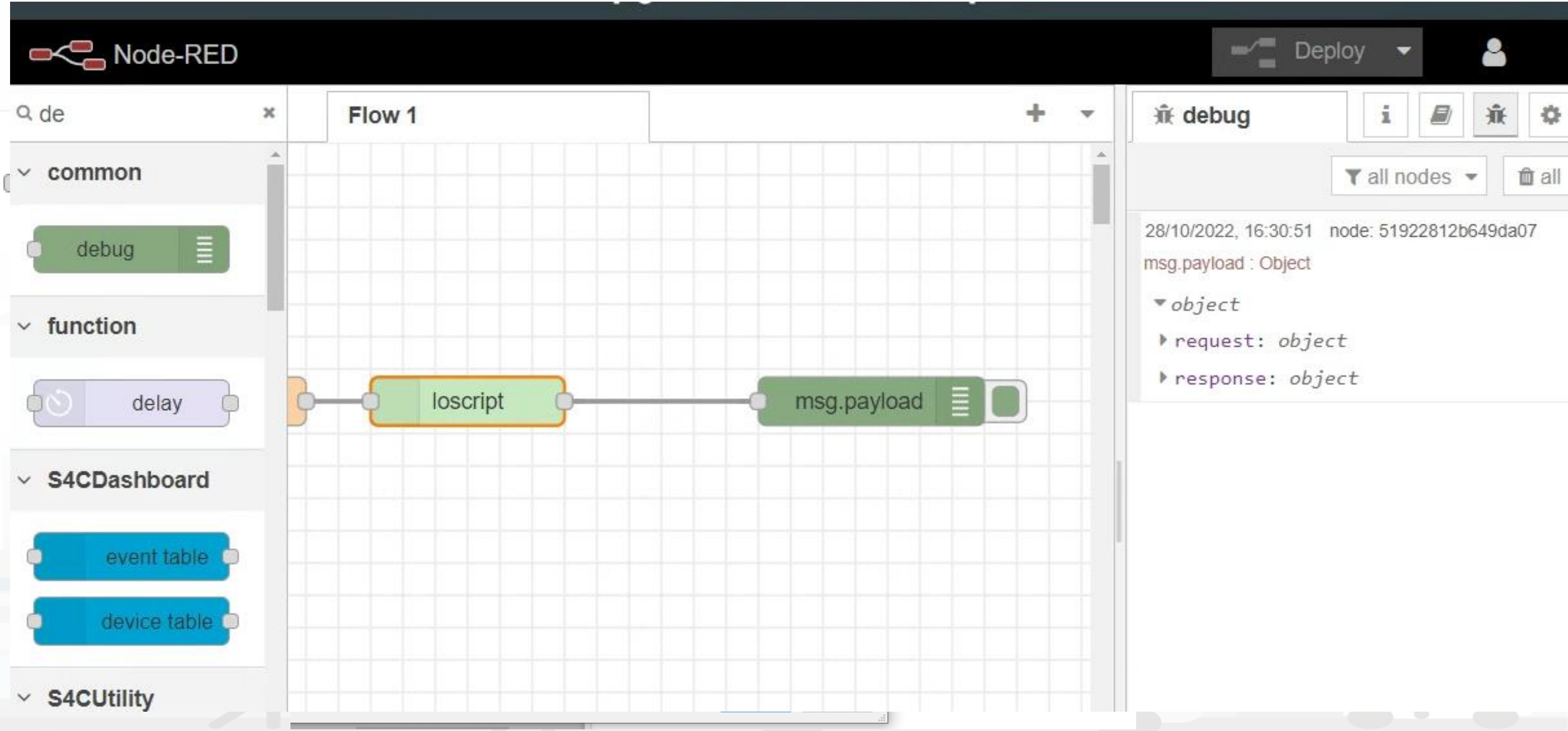
Back to the IoTApp



Back to the IoTApp






Node-RED
Deploy
de
Flow 1
debug
all nodes
all
common
debug
function
delay
S4CDashboard
event table
device table
S4CUtility
loscript
msg.payload
28/10/2022, 16:30:51 node: 51922812b649da07
msg.payload : Object
object
request: object
response: object

UNIVERSITÀ DEGLI STUDI FIRENZE | DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE | DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB | SNAP4CITY

Private Device Data Retrieval

- We'll use the cloud installation of jupyterhub
- https://www.snap4city.org/drupal/node/650

Not All The Device in Snap4City are public...

for some you'll need an access token to the private **IoT Device** of that authenticated user ¶

so let's get the username and password

[1]:

```
### in the config.py file that i've created are stored the user and password for the snap4city authentication
# snap4cityauth = dict(
#     user = 'user name of snap4city',
#     psw = 'the password of the user',
#     clid= '<client id depending on the App kind>' has to be obtained from Snap4City organization by sending an email to snap4city@disit.org.
# )
import config
utente = config.snap4cityauth['user']
password = config.snap4cityauth['psw']
client_id = config.snap4cityauth['clid']
```

Private Device Data Retrieval

next let's get the auth token ¶

2]:
```
import requests
import json
url = "https://www.snap4city.org/auth/realms/master/protocol/openid-connect/token/"
data = {"client_id": client_id,"grant_type":"password","username":utente,"password":password}
r=requests.post(url, data)
print(r.status_code, r.reason)
responseToken=json.loads(r.text)
refreshToken = responseToken["refresh_token"]
print("access_token : {}... expires in {}s, token_type: {}".format(responseToken['access_token'][:20],responseToken['expires_in'],responseToken['token_type'] ))

#to update the token using the refresh_token
url = "https://www.snap4city.org/auth/realms/master/protocol/openid-connect/token/"
data = {"client_id": client_id,"grant_type":"refresh_token","scope":"openid profile","refresh_token":refreshToken}
r=requests.post(url, data)
print("updating token using the refresh token ",r.status_code, r.reason)
responseToken=json.loads(r.text)
```

```
200 OK
access_token : eyJhbGciOiJSUzI1NiIs... expires in 1500s, token_type: bearer
updating token using the refresh token  200 OK
```

Private Device Data Retrieval

so now you can access the private iot device data...

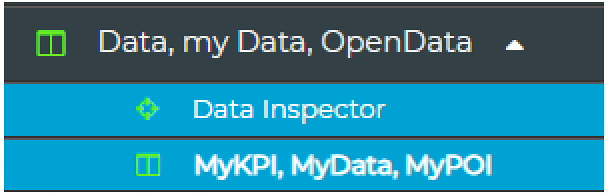
```
]: auth_token=responseToken['access_token']
hed = {'Authorization': 'Bearer ' + auth_token}

url = "https://www.snap4city.org/superservicemap/api/v1?serviceUri=http://www.disit.org/km4city/resource/iot/orionUNIFI/DISIT/118907.682_485819.390-Plastic&accessToke

response = requests.get(url, headers=hed)
if response.status_code == 200: # ok
    print(json.loads(response.text))
```

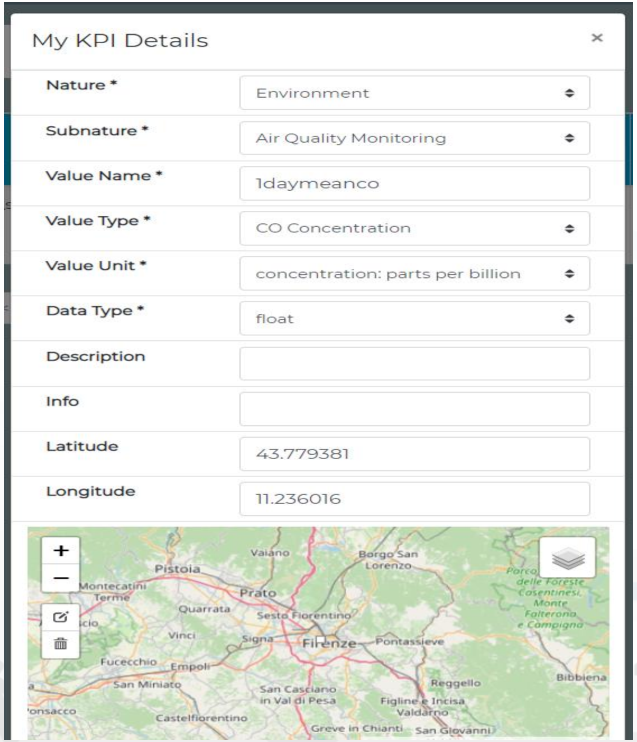
{'Service': {'features': [{'geometry': {'coordinates': [4.857379, 52.359085], 'type': 'Point'}, 'properties': {'address': '', 'avgStars': 0, 'brokerName': 'orionUNIFI', 'cap': '', 'city': '', 'civic': '', 'comments': [], 'description': 'Plastic', 'email': '', 'fax': '', 'format': 'json', 'frequencySec': '600', 'isMobile': '', 'linkDBpedia': [], 'macaddress': '', 'maintenanceUrl': '', 'maxCapacity': '5', 'minCapacity': '', 'model': 'AmsterdamPlasticContainer', 'multimedia': '', 'name': '118907.682_485819.390-Plastic', 'nature': 'Environment', 'organization': 'DISIT', 'ownership': '', 'phone': '', 'photoOrigs': [], 'photoThumbs': [], 'photos': [], 'producer': 'Amsterdam city', 'protocol': 'ngsi', 'province': '', 'realtimeAttributes': {'dateObserved': {'attr_type': 'DeviceAttribute', 'data_type': 'string', 'different_values': '0', 'value_bounds': 'unspecified', 'value_refresh_rate': '300', 'value_type': 'timestamp', 'value_unit': 'timestamp'}, 'weight': {'attr_type': 'DeviceAttribute', 'data_type': 'float', 'different_values': '0', 'value_bounds': 'unspecified', 'value_refresh_rate': '300', 'value_type': 'weight', 'value_unit': 'Kg'}}, 'serviceType': 'Environment_Waste_container', 'serviceUri': 'http://www.disit.org/km4city/resource/iot/orionUNIFI/DISIT/118907.682_485819.390-Plastic', 'starsCount': 0, 'subnature': 'Waste_container', 'typeLabel': 'Waste container', 'website': '', 'wktGeometry': ''}, 'type': 'Feature'}], 'type': 'FeatureCollection'}, 'realtime': {'head': {'vars': ['measuredTime', 'dateObserved', 'weight']}, 'results': {'bindings': [{'dateObserved': {'value': '2022-01-14T09:52:09.000Z'}, 'measuredTime': {'value': '2022-01-14T10:52:09.000+01:00'}, 'weight': {'value': '120'}}]}}}

MyKPI Data Analytics to Dashboard

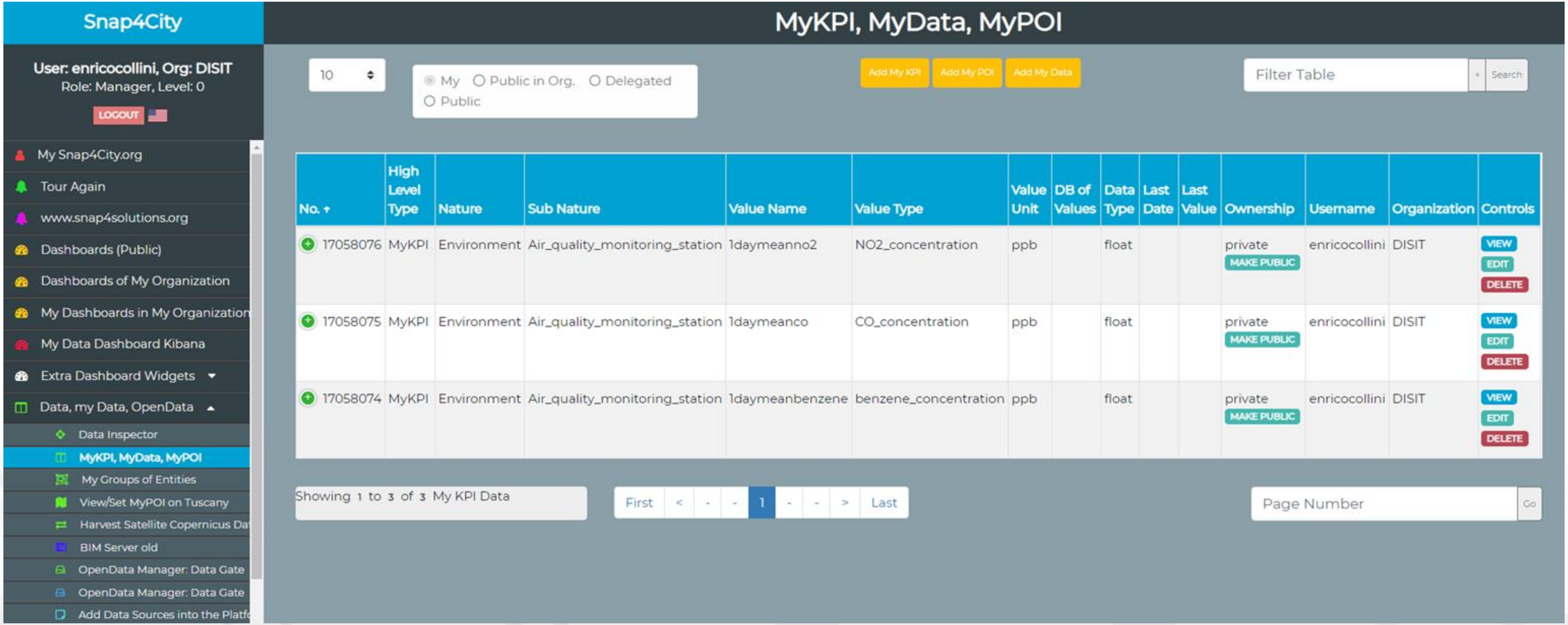
- Create 1day myKpis
- Modify the node-red script to get 1 day of data or
  - benzene
  - no2
  - co
  - and add the data to the kpis
- Create the real time time series dashboard of the 1day mean values of pollutants

Create 1day myKpis





Create 1day myKpis



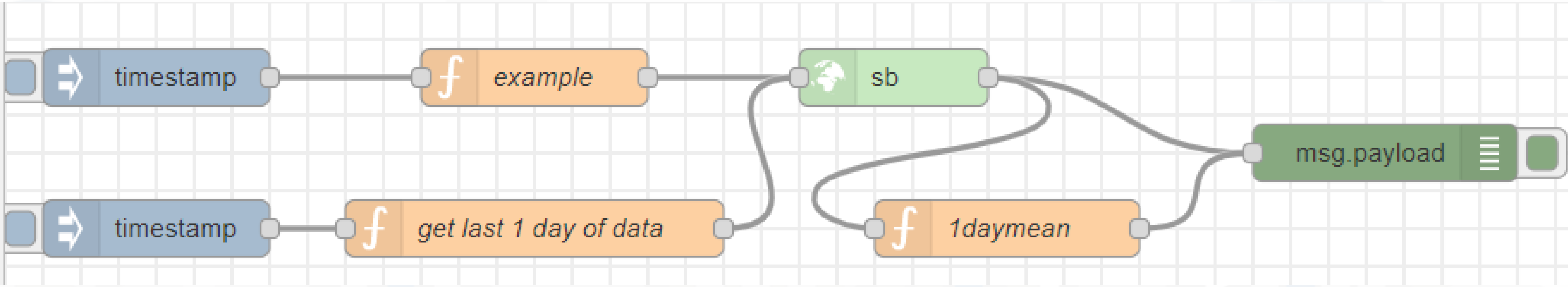
No.	High Level Type	Nature	Sub Nature	Value Name	Value Type	Value Unit	DB of Values	Data Type	Last Date	Last Value	Ownership	Username	Organization	Controls
17058076	MyKPI	Environment	Air_quality_monitoring_station	1daymeanno2	NO2_concentration	ppb		float			private MAKE PUBLIC	enricocollini	DISIT	VIEW EDIT DELETE
17058075	MyKPI	Environment	Air_quality_monitoring_station	1daymeanco	CO_concentration	ppb		float			private MAKE PUBLIC	enricocollini	DISIT	VIEW EDIT DELETE
17058074	MyKPI	Environment	Air_quality_monitoring_station	1daymeanbenzene	benzene_concentration	ppb		float			private MAKE PUBLIC	enricocollini	DISIT	VIEW EDIT DELETE

Modify the node-red script to get 1 day of data or

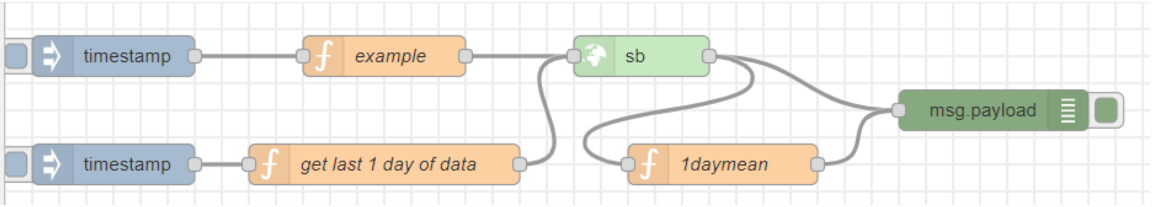
benzene

no2

co



UNIVERSITÀ DEGLI STUDI FIRENZE | DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE | DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB | SNAP4CITY



2/11/2022, 14:14:18 node: 51922812b649da07

msg.payload : Object

▾object

benzene: 2.1695652173913045

no2: 41.69565217391305

co: 0.6043478260869566

Edit function node

Delete | Cancel | Done

Properties

Name: get last 1 day of data

Setup | On Start | **On Message** | On Stop

```
let ora = new Date();
var oraprima = new Date();
oraprima.setHours(ora.getHours() - 24);
msg.payload = {
    "start_date":oraprima.toISOString().slice(0,19),
    "end_date":ora.toISOString().slice(0,19),
    "sensor_uri":"http://www.disit.org/km4city/resource/iot/orionUNIFI/D
}
return msg;
```

Edit function node

Delete | Cancel | Done

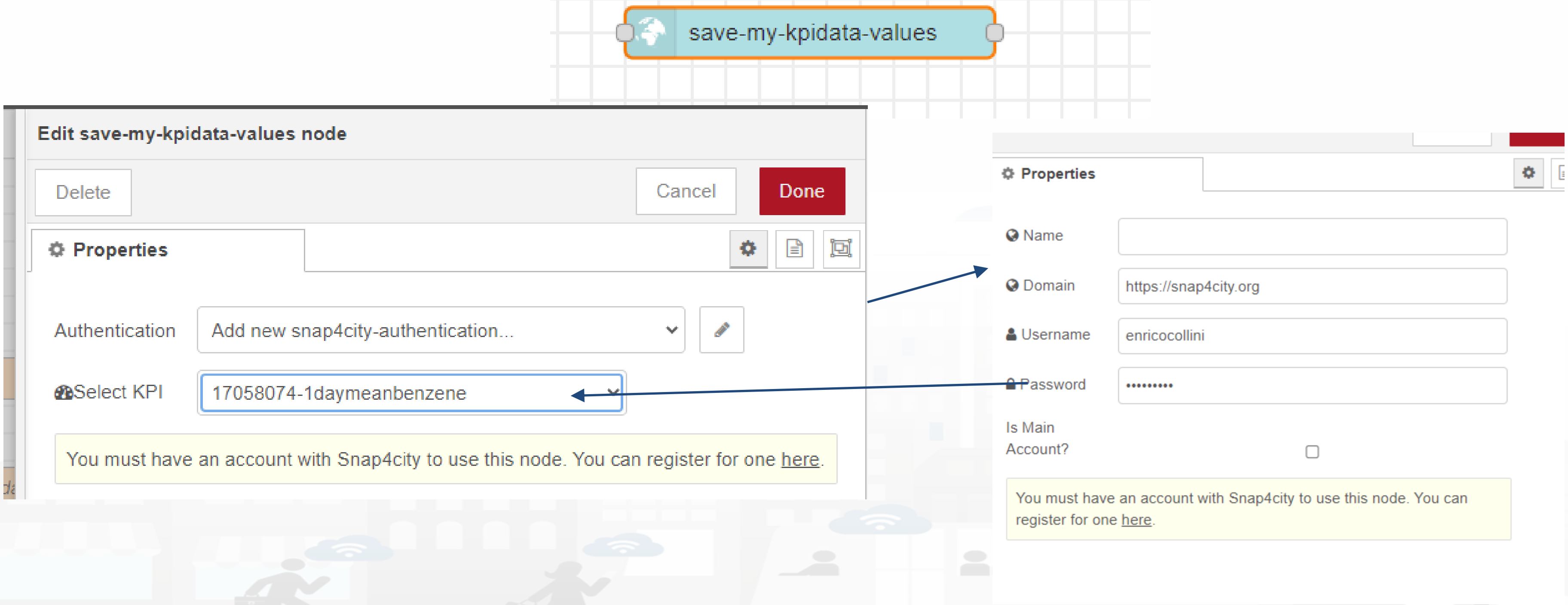
Properties

Name: 1daymean

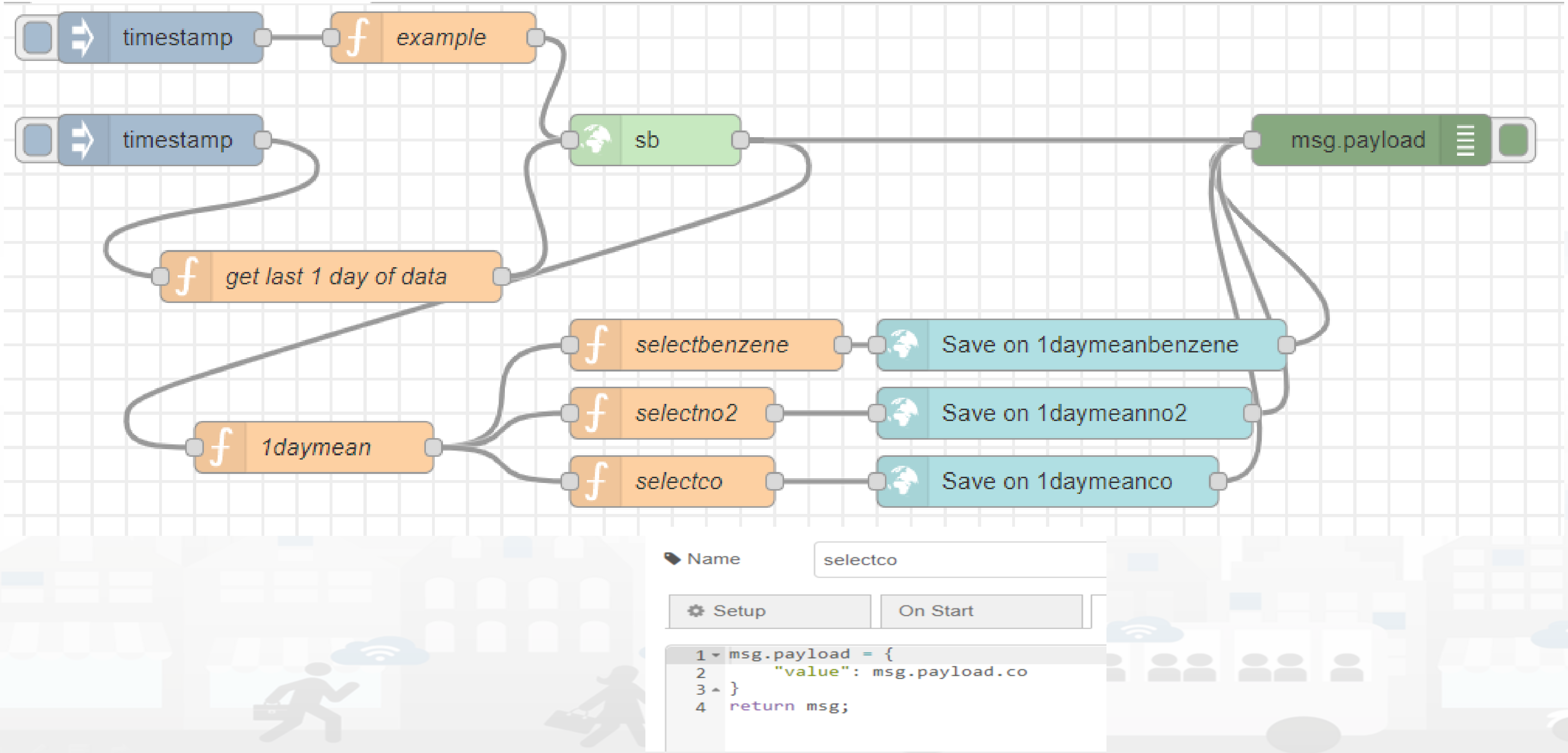
Setup | On Start | **On Message** | On Stop

```
let valori =  msg.payload.response.realtime.results.bindings;
benzene = 0;
no2= 0
co = 0
count = 0
for(let i=0; i<valori.length; i++){
    benzene += parseFloat(valori[i].Benzene.value);
    no2 += parseFloat(valori[i].NO2.value);
    co += parseFloat(valori[i].CO.value);
    count +=1
}
msg.payload = {
    "benzene" : benzene/count,
    "no2":no2/count,
    "co":co/count
}
return msg;
```

and add the data to the kpis



and add the data to the kpis



check the data of the kpis

MyKPI, MyData, MyPOI

10 | My | Public in Org. | Delegated | Public

Add My KPI | Add My POI | Add My Data

Filter Table | Search

No. ↑	High Level Type	Nature	Sub Nature	Value Name	Value Type	Value Unit	DB of Values	Data Type	Last Date	Last Value	Ownership	Username
17058076	MyKPI	Environment	Air_quality_monitoring_station	1daymeanno2	NO2_concentration	ppb		float	2/11/2022, 14:25:23	41.69565217391305	private MAKE PUBLIC	enricocollini

Organization DISIT

Controls VIEW EDIT DELETE

Data VALUES METADATA KIBANA

Visibility DELEGATE USERS DELEGATE ORGANIZATIONS CHANGE OWNERSHIP

No. ↑	High Level Type	Nature	Sub Nature	Value Name	Value Type	Value Unit	DB of Values	Data Type	Last Date	Last Value	Ownership	Username
17058075	MyKPI	Environment	Air_quality_monitoring_station	1daymeanco	CO_concentration	ppb		float	2/11/2022, 14:25:23	0.6043478260869566	private MAKE PUBLIC	enricocollini
17058074	MyKPI	Environment	Air_quality_monitoring_station	1daymeanbenzene	benzene_concentration	ppb		float	2/11/2022, 14:25:23	2.1695652173913045	private MAKE PUBLIC	enricocollini

check the data of the kpis

MyKPI, MyData, MyPOI

Return to My KPI Data List

Refresh

Values of KPIData: **No.** 17058076 **Nature** Environment | **Sub Nature** Air_quality_monitoring_station | **Value Name** 1daymeanno2 | **DB of Values**

10

Add New My KPI Value

CLEAR ALL VALUES

Filter Table × Search

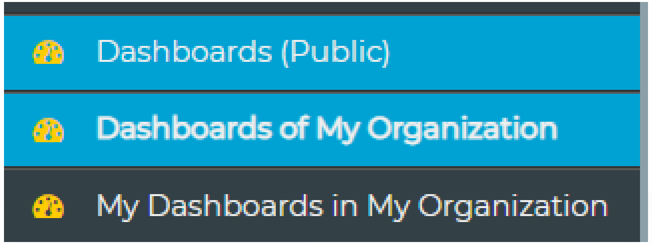
No.	Value	Latitude	Longitude	Data Time ↑	Insert Time	Controls
25692466	41.69565217391305			2/11/2022, 14:25:23	2/11/2022, 14:25:24	EDIT DELETE
25692462	41.69565217391305			2/11/2022, 14:25:19	2/11/2022, 14:25:19	EDIT DELETE
25692422	41.69565217391305			2/11/2022, 14:23:24	2/11/2022, 14:23:26	EDIT DELETE

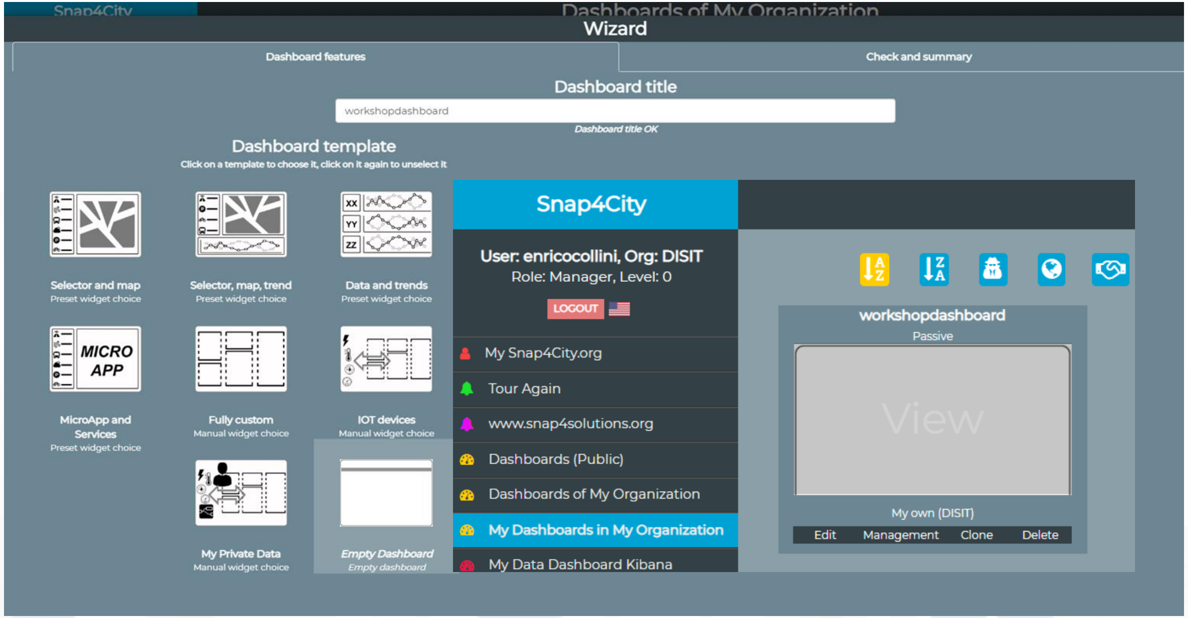
Showing 1 to 3 of 3 My KPI Value

First < - - 1 - - > Last

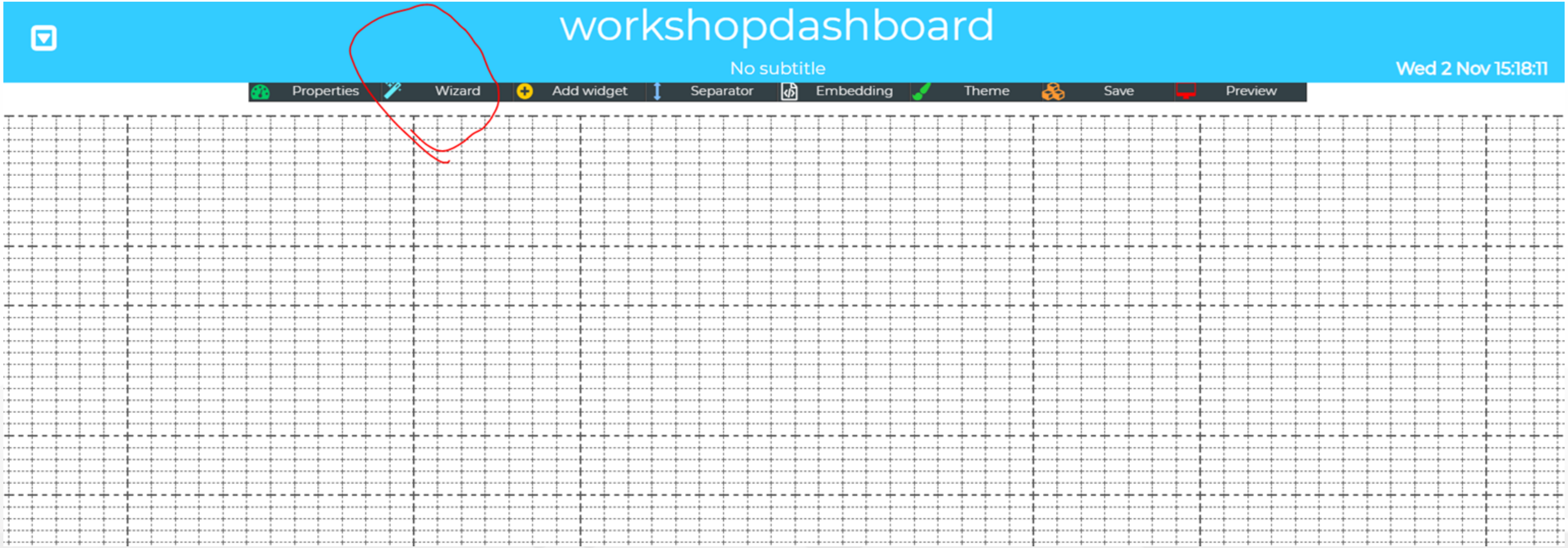
Page Number Go

Create the real time time series dashboard of the 1day mean values of pollutants

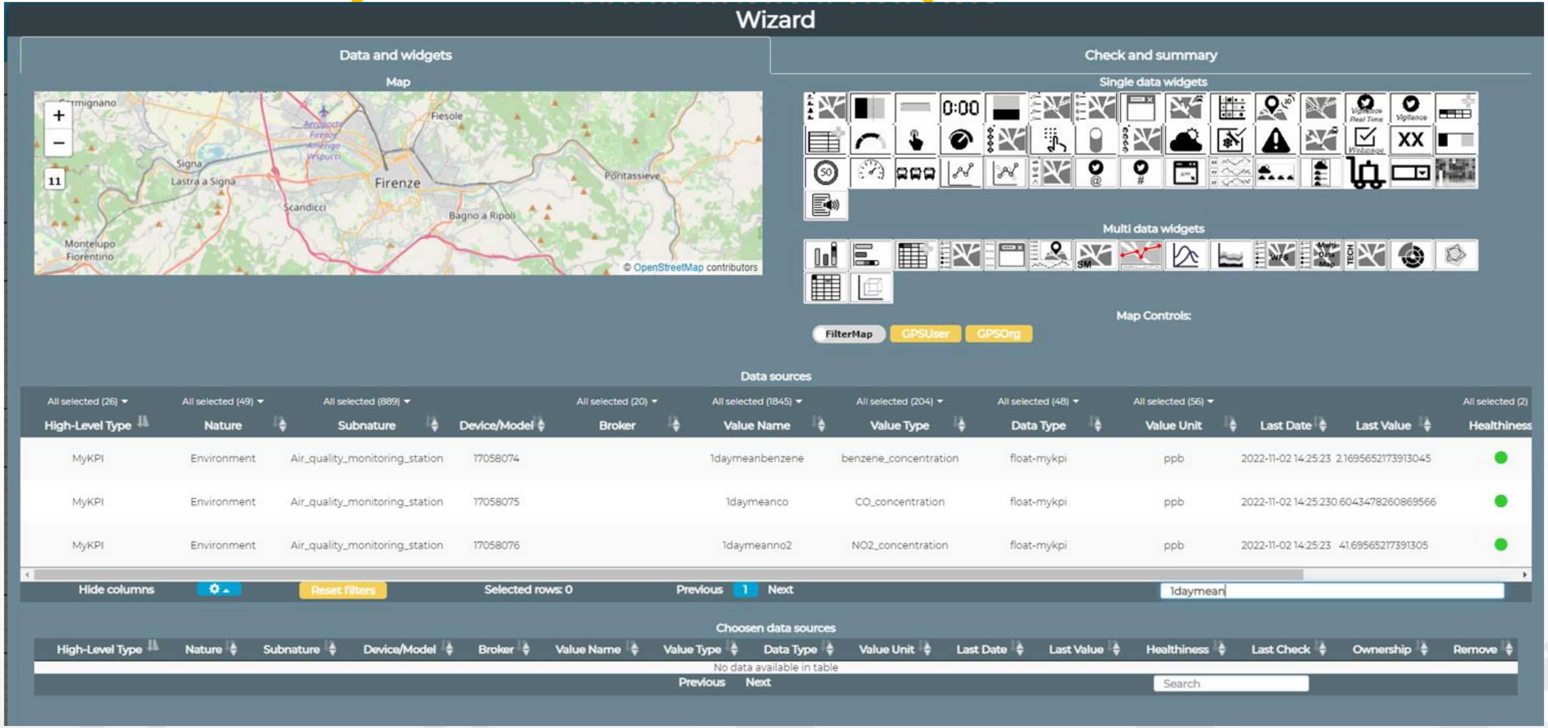




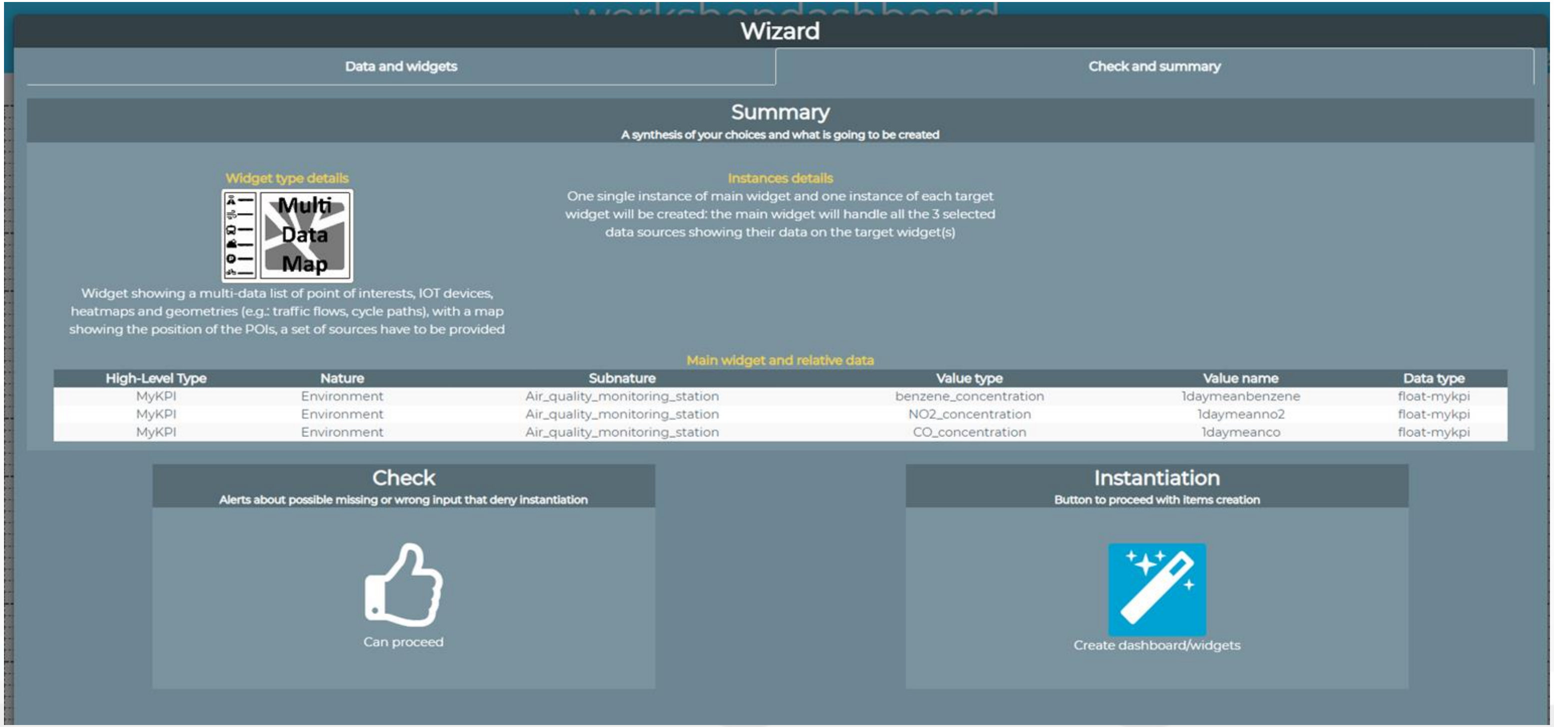
Create the real time time series dashboard of the 1day mean values of pollutants



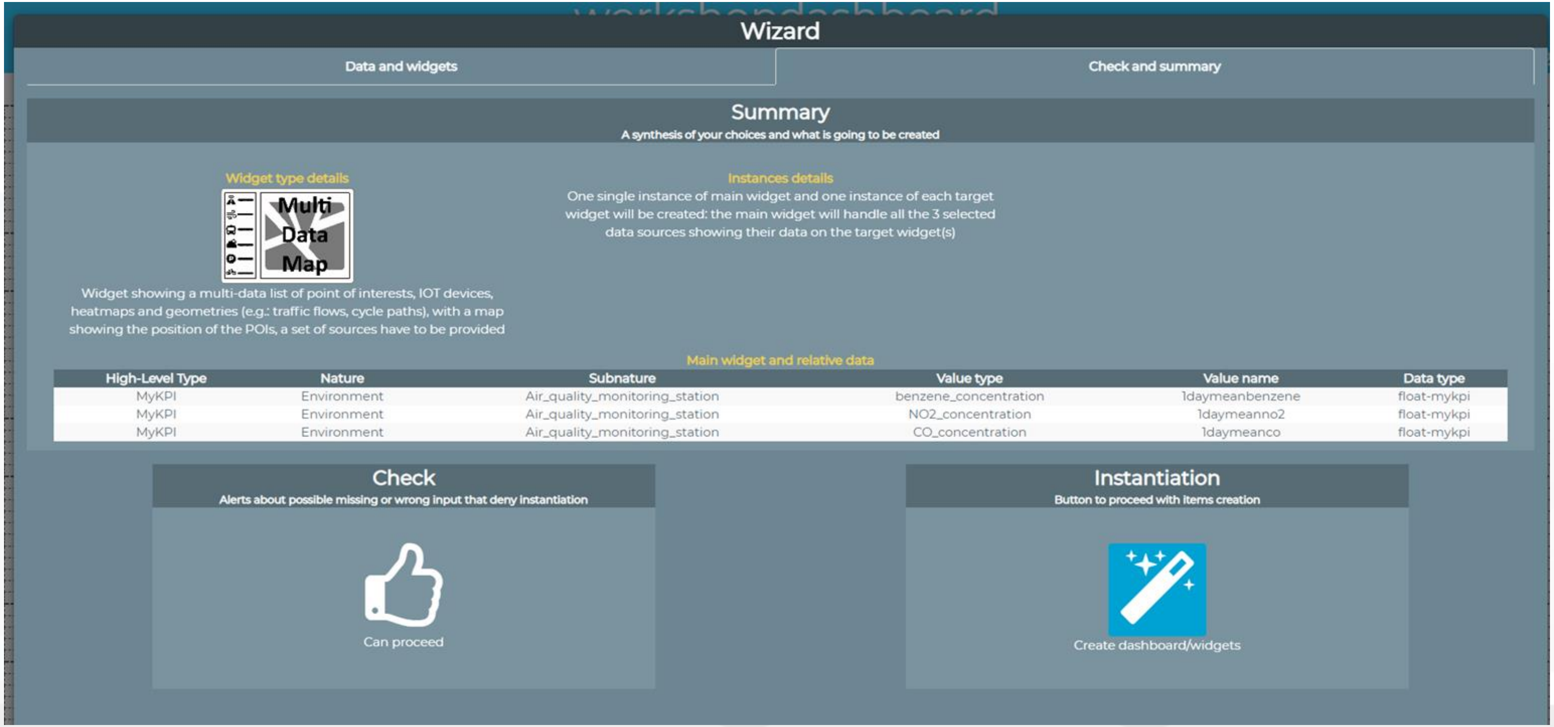
Create the real time time series dashboard of the 1day mean values of pollutants



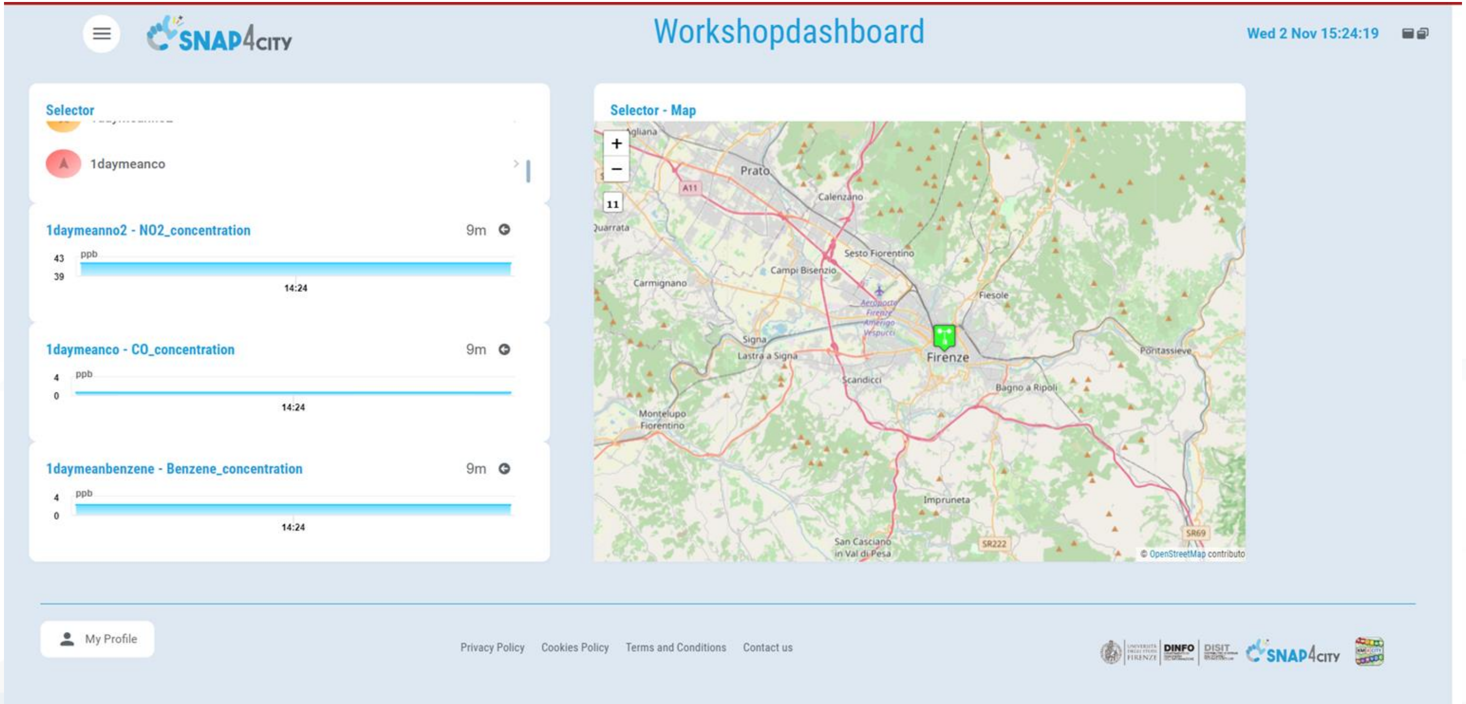
Create the real time time series dashboard of the 1day mean values of pollutants



Create the real time time series dashboard of the 1day mean values of pollutants



High-Level Type	Nature	Subnature	Value type	Value name	Data type
MyKPI	Environment	Air_quality_monitoring_station	benzene_concentration	1daymeanbenzene	float-mykpi
MyKPI	Environment	Air_quality_monitoring_station	NO2_concentration	1daymeanno2	float-mykpi
MyKPI	Environment	Air_quality_monitoring_station	CO_concentration	1daymeanco	float-mykpi


SNAP4CITY
Workshopdashboard
Wed 2 Nov 15:24:19
Selector
1daymeanco
1daymeanno2 - NO2_concentration
9m
ppb
43
39
14:24
1daymeanco - CO_concentration
9m
ppb
4
0
14:24
1daymeanbenzene - Benzene_concentration
9m
ppb
4
0
14:24
Selector - Map
Prato
Calenzano
Sesto Fiorentino
Campi Bisenzio
Carmignano
Fiesole
Signa
Lastra a Signa
Firenze
Scandicci
Bagno a Ripoli
Pontassieve
Montelupo Fiorentino
Impruneta
San Casciano in Val di Pesa
© OpenStreetMap contributo
My Profile
Privacy Policy
Cookies Policy
Terms and Conditions
Contact us