Big Data Architectures

Time Series & Optimizations
4 Smart City Applications

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

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 irregular (events)

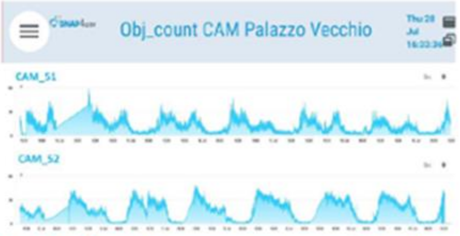
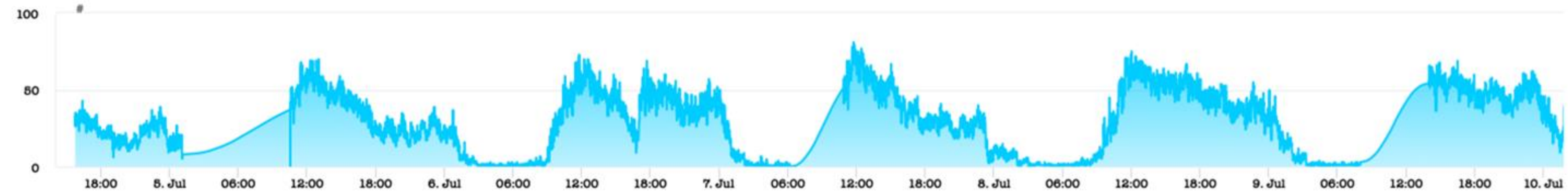
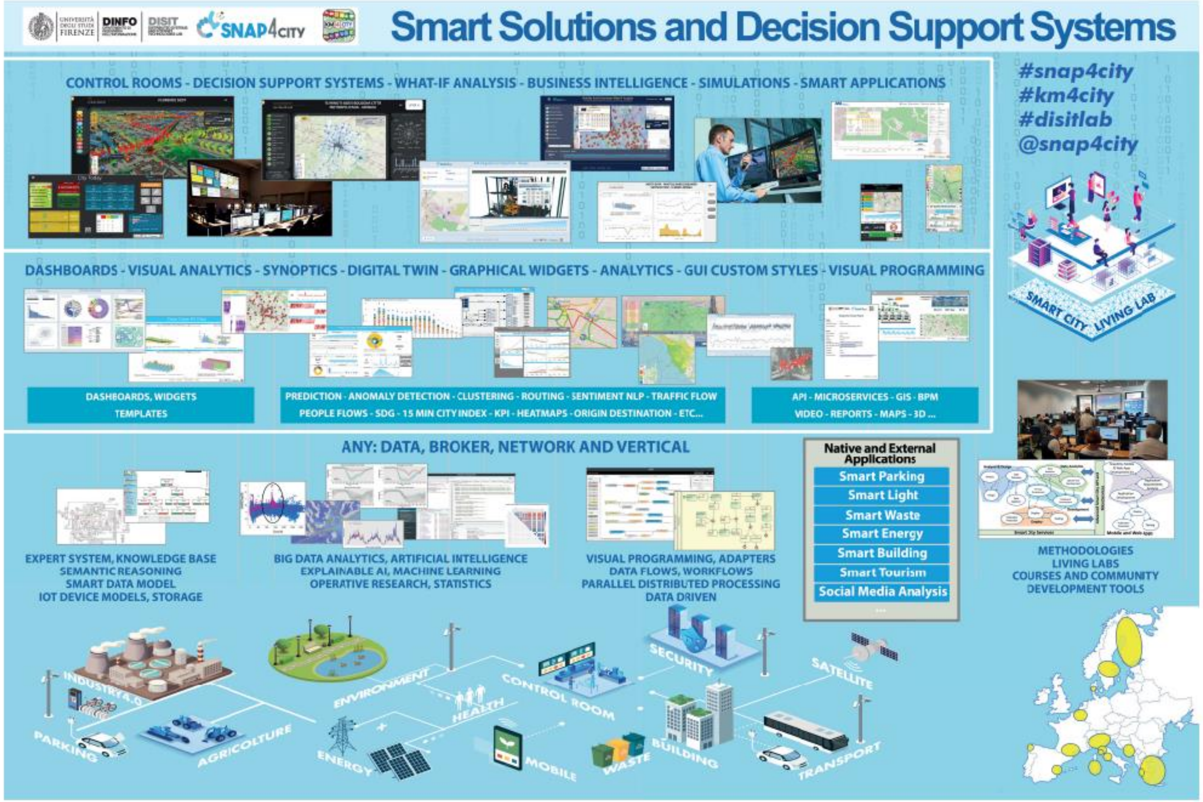


FIGURE 11. Monitoring Dashboard of people counting in Piazza Della Signoria, Florence



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



Time Series Data Analysis (TSA)

What can you do with time-series data?

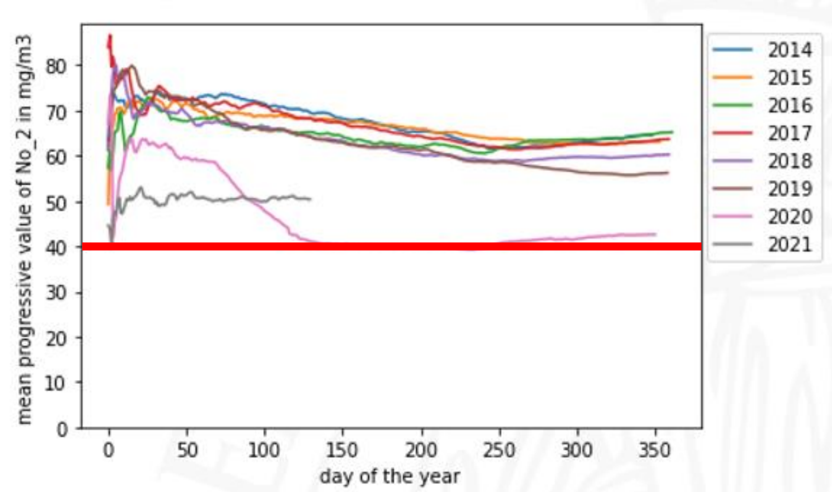
Analyze change (past - present -future)

3 main analysis types:

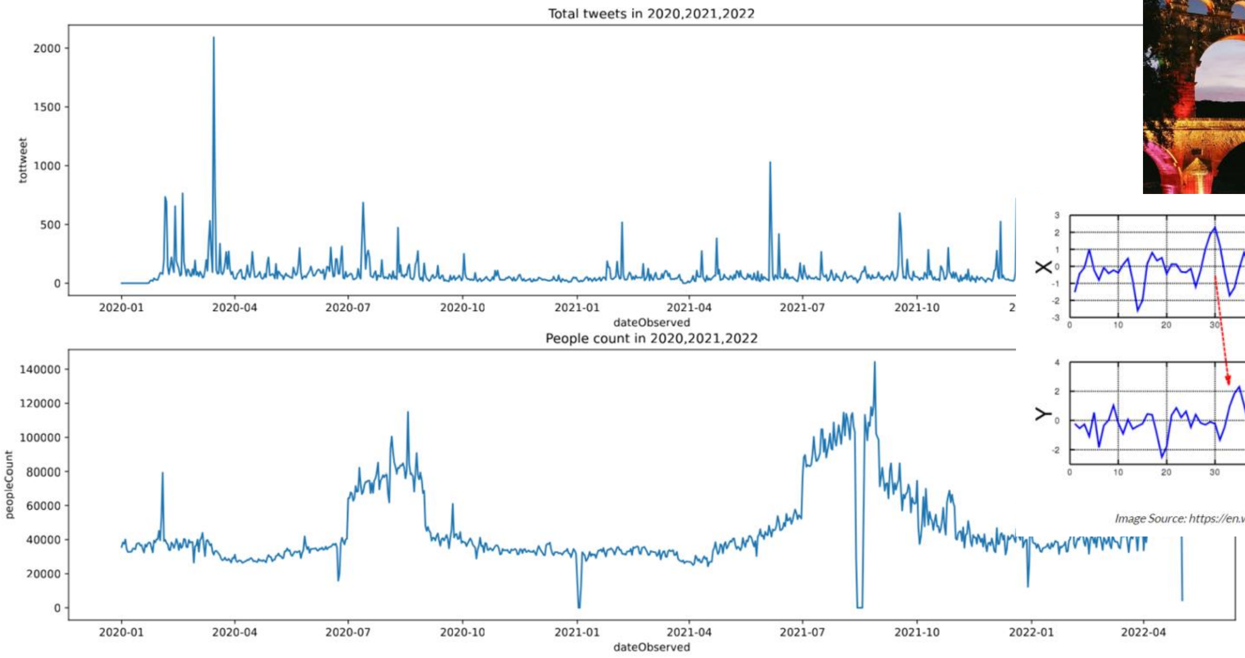
A) Access the impact of a single event (descriptive)
B) Study the interaction between a set of values
C) Forecast Future Values of a Time-Series using the previous values of one series (or also values from others) (prediction)

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



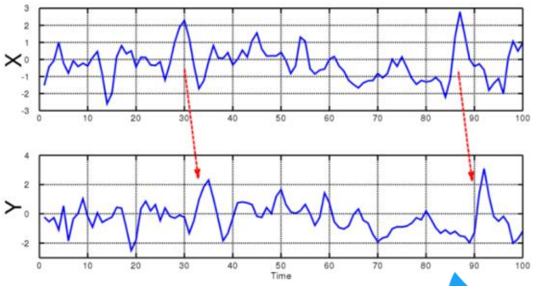
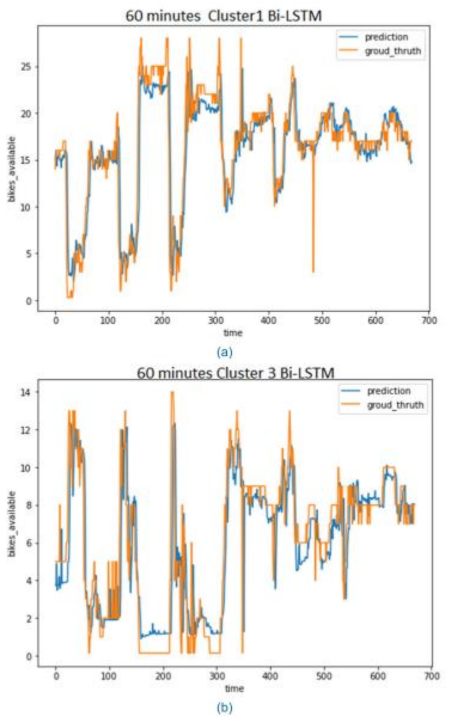
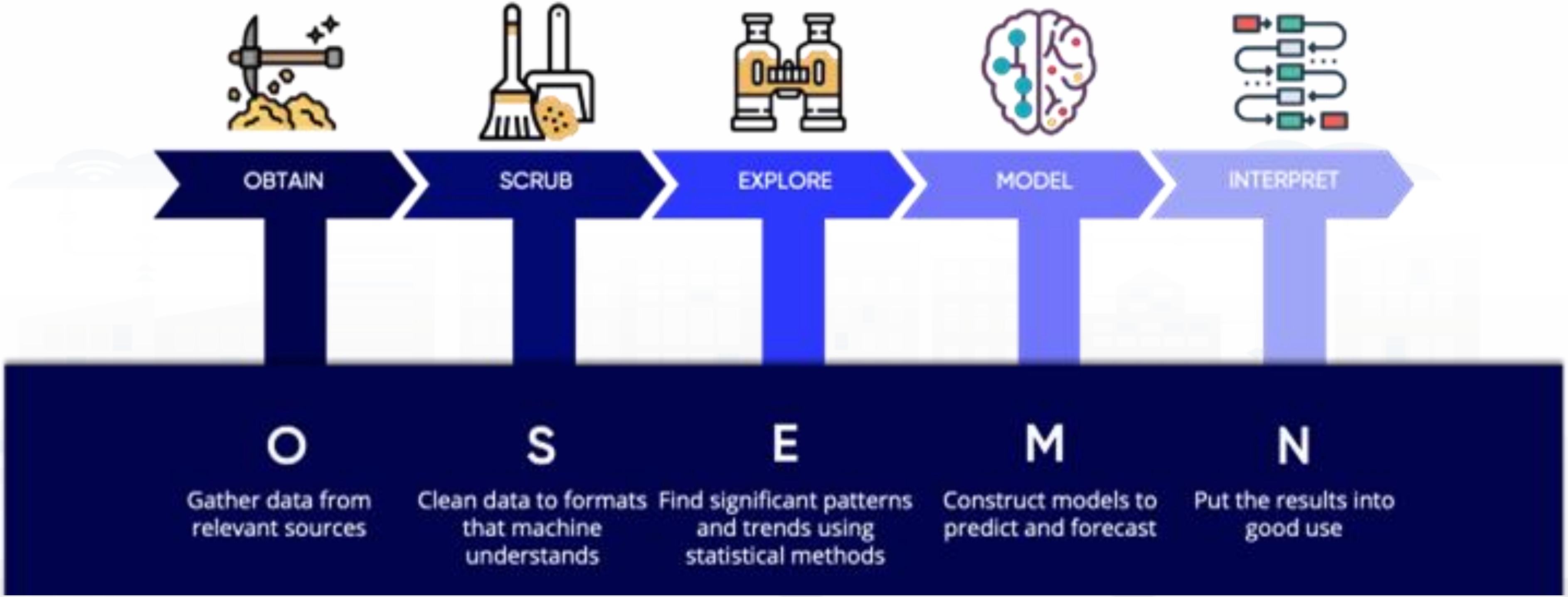


Image Source: https://en.wikipedia.org/wiki/Granger_causality

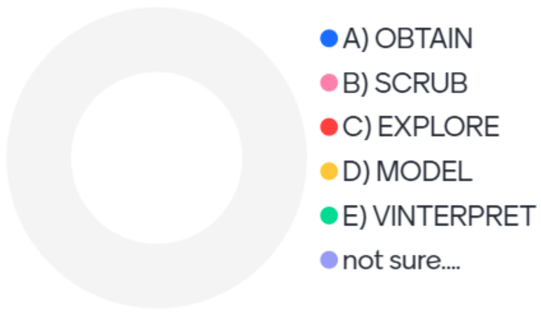
C) Forecast Future Values of a Time-Series



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**:

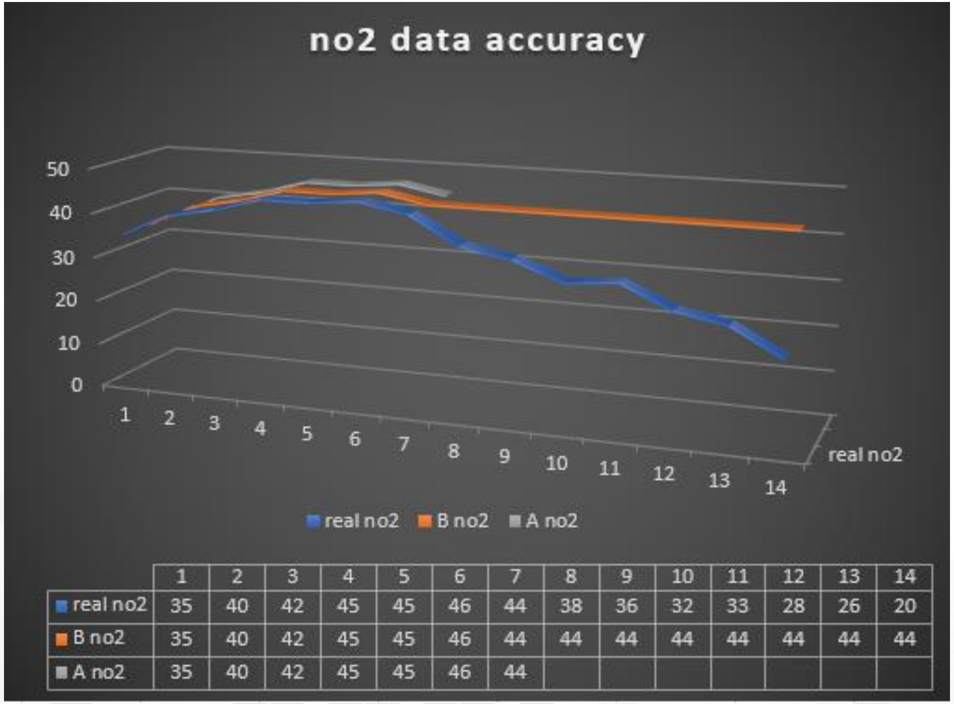
- are the collected data correct / do they accurately represent what it should

Data Acquisition…

- IoT environment sensor with air pollutants breaks.

A) keep sending the last value

B) sends the data only if available

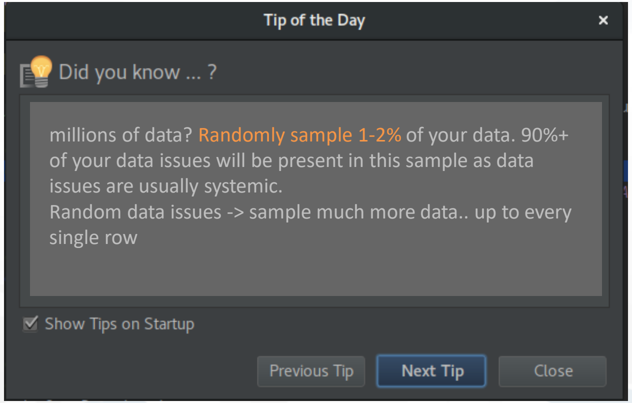


	1	2	3	4	5	6	7	8	9	10	11	12	13	14
real no2	35	40	42	45	45	46	44	38	36	32	33	28	26	20
B no2	35	40	42	45	45	46	44	44	44	44	44	44	44	44
A no2	35	40	42	45	45	46	44							

Scrub Data - Information Quality Pillars

- **Validity:** data really measure measure what is intended?

- **Timely**: data should be received in order and depending on the application really fast!

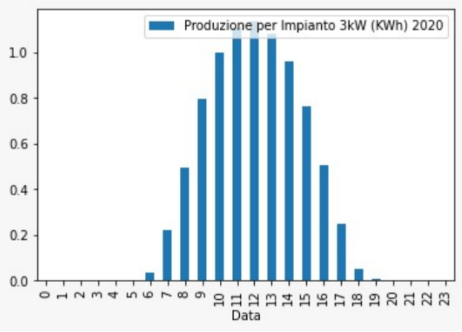


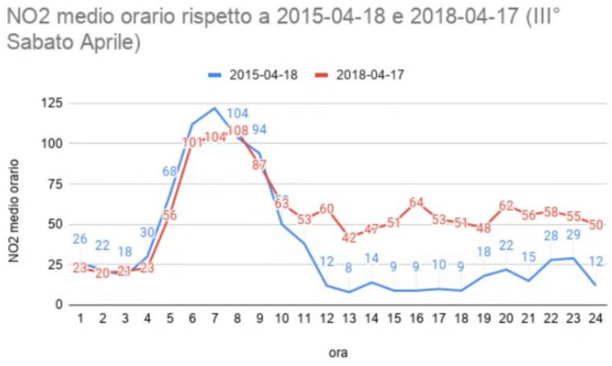
Explore Data

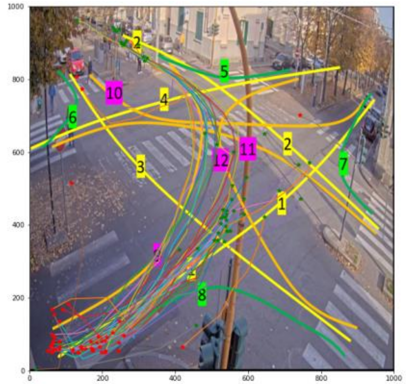
Once your data is ready to be used, and right before you jump into AI and Machine Learning, you will have to examine the data.

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

- Distributions
- Patterns / Trends
- Clustering







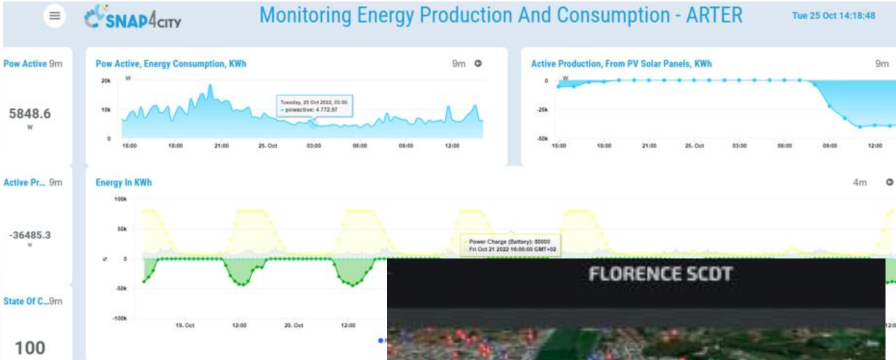
Model

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

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



Road to Time Series Forecasting

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

Forecasting Methods Selection

Mathematical Formulation of Time Series

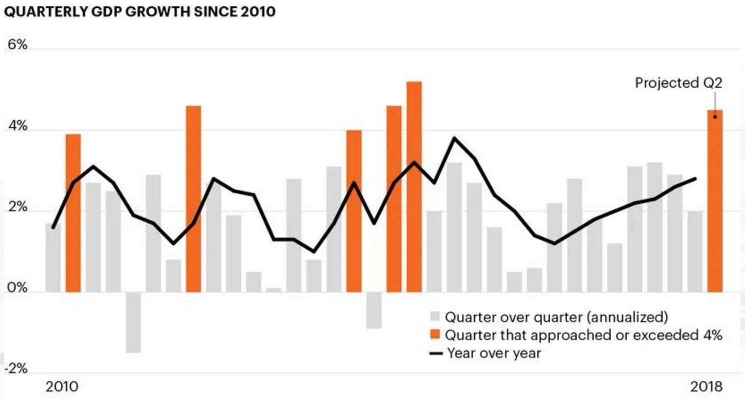
Time Series is the set of several observations of a phenomenon with respect to time.

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).



Source: Macrobond, based on data from the U.S. Bureau of Economic Analysis, as of July 19, 2018. Q2 2018 GDP data is a projection based on the current GDPNow data from the Federal Reserve Bank of St. Louis.

Moments of Time Series

The "moments" of a time series are statistical measures that describe various characteristics of the distribution of the observed data over time. These moments are important for analyzing the central tendency, variability, and correlations present in the time series.

1. Mean

The **mean** is the first moment of the distribution of data. It represents the central tendency of the time series, i.e., the average value around which the data tend to cluster.

For a time series y_t with T observations (where t represents the different time instants):

$$\text{Mean} = \mu = \frac{1}{T} \sum_{t=1}^{T} y_t$$

Moments of Time Series

The "moments" of a time series are statistical measures that describe various characteristics of the distribution of the observed data over time. These moments are important for analyzing the central tendency, variability, and correlations present in the time series.

2. Variance

The **variance** is the second moment of the distribution and measures the dispersion of the data around the mean. It indicates how much the values of the time series deviate from the mean.

The formula for variance is:

$$\text{Variance} = \sigma^2 = \frac{1}{T}\sum_{t=1}^{T}(y_t - \mu)^2$$

Moments of Time Series

3. Autocovariance

Autocovariance measures the correlation between values of a time series at different time points. In other words, it tells us how much two values separated by a certain time lag are related to each other.

The autocovariance between values y_t and y_{t+h} (where h is the lag, or the time distance between the two observati

$$\text{Autocovariance}(h) = \frac{1}{T-h}\sum_{t=1}^{T-h}(y_t-\mu)(y_{t+h}-\mu)$$

In practice, autocovariance tells us how related the values of the series are over time (at different lags). If the autocovariance is positive, it means that the two values tend to move in the same direction; if it's negative, they tend to move in opposite directions.

Moments of Time Series

Practical Example:

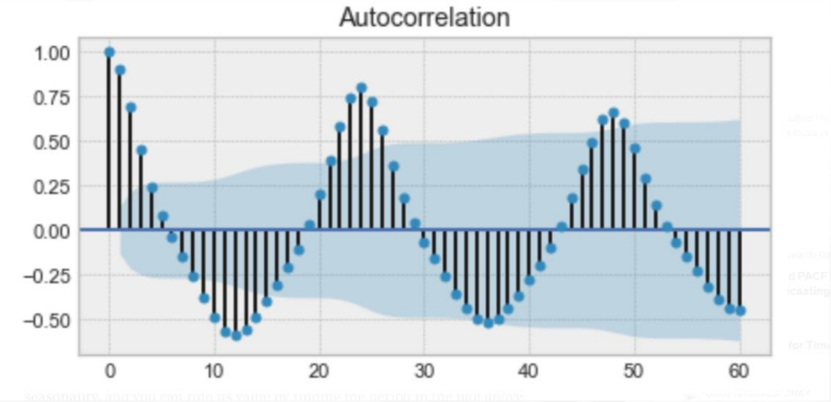
Suppose you have quarterly GDP data for a country over several years, as in your example, and you want to calculate the variance or autocovariance of this GDP. If you calculate the **mean** of the GDP values, you get the average growth rate of the GDP for the considered period. The **variance** tells you how much the GDP values deviate from this average, i.e., how volatile the GDP is. The **autocovariance** helps you understand whether the GDP in one quarter is correlated with the GDP in another quarter. For example, whether an increase in GDP in one quarter is followed by an increase in the next quarter.

https://colab.research.google.com/drive/1s9N_bGmAwWyeLiOh-9GSRdLH5IJcpyls?usp=sharing

Time Series Characteristics

$$\rho_k = \frac{\mathrm{Cov}(X_t, X_{t-k})}{\mathrm{Var}(X_t)}$$

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. $Y_1, Y_2, Y_3, \dots, Y_{T-2}, Y_{T-1}, Y_T$
- 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}	$\dots$	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

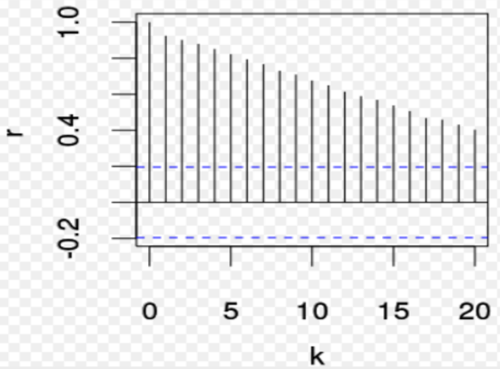
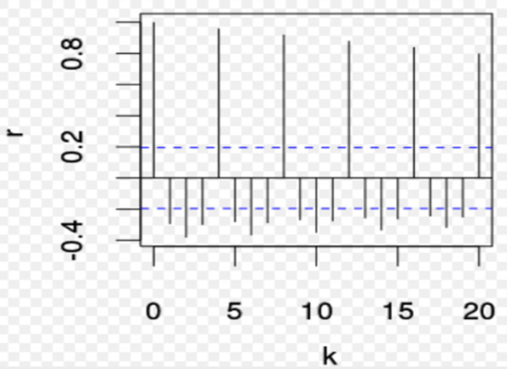
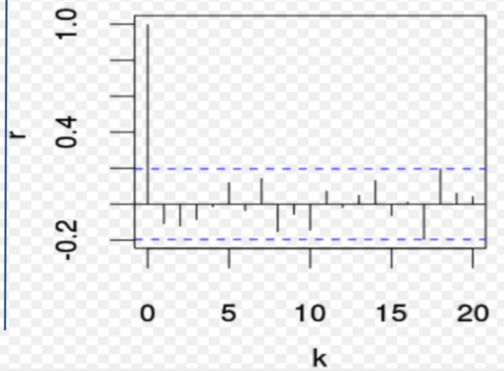
Understanding ACF Plots

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} :

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

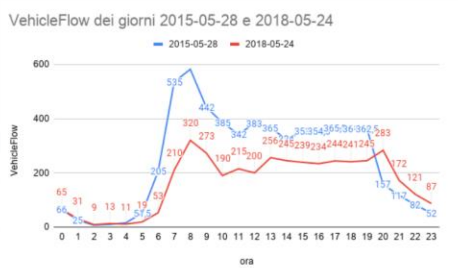
The autocovariance divided by the product of the standard deviations, i.e. the variance

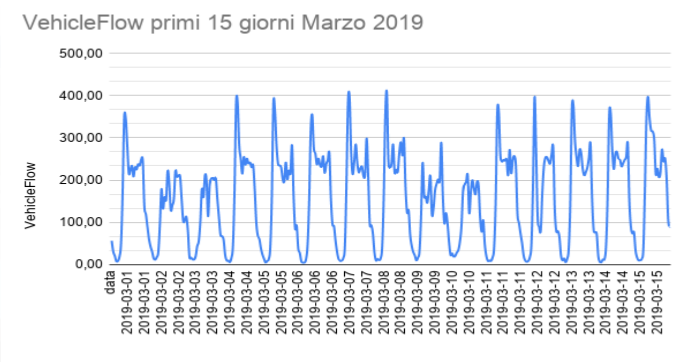
Original Time Series characteristics

linear trend Behaviour	Seasonal component	Stochastic
		

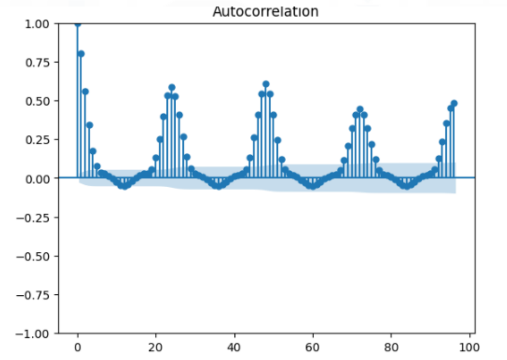
Time Series Characteristics

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.

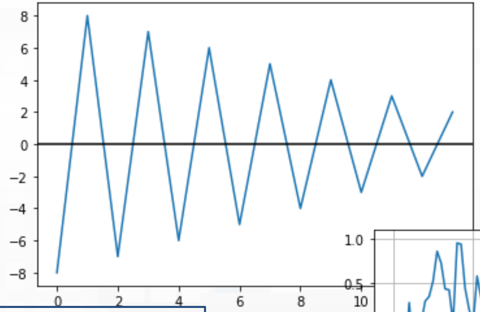


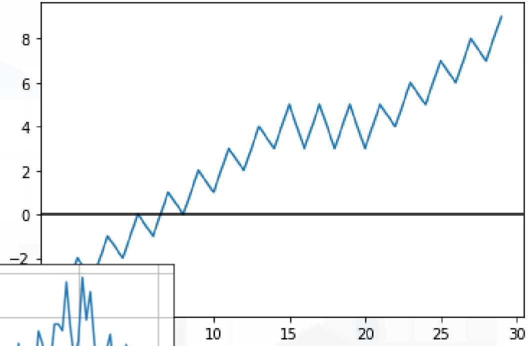
Time Series Characteristics

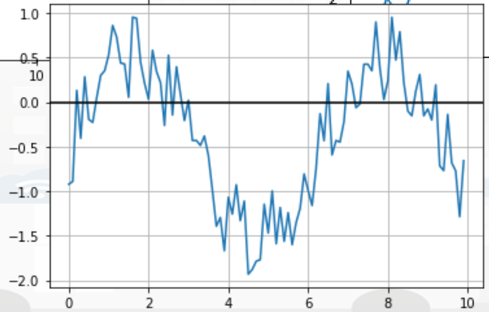
Stationarity is an important characteristic of time series that the majority of statistical forecasting techniques require. A time series is said to be stationary if its statistical properties do not change over time and there is not seasonality…

3 requirements:

- μ const
- σ^2 const
- No Seasonality







How to check for stationarity:

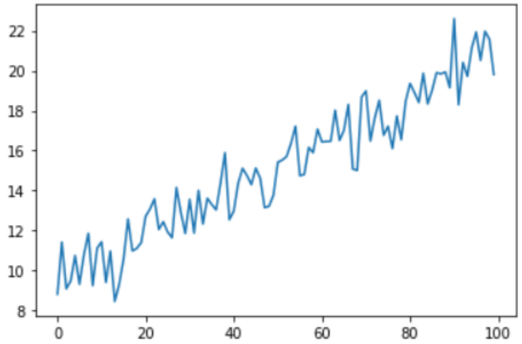
1) Visually as we did
2) Global mean vs local mean ….
3) **Statistical Tests**

Making a Time Series Stationary

```python
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 $D_t = Y_t - Y_{t-1} =$

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

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

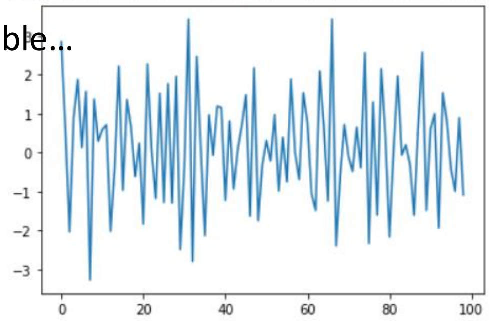
$$\underbrace{b_1 + (\epsilon_t - \epsilon_{t-1})}_{\text{const} \quad \text{indip vars}}$$

μ const b_1

σ^2 k^2 k^2 = $2k^2$ const

```python
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

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

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 example

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

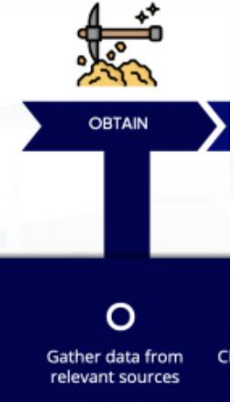
- Producing short-term 1h predictions of:
  - (i) number of bikes available in bike-sharing systems stations,
  - (ii) free slots.
- Identify the best solution among different AI/ML Techniques.
- Understand which are the most relevant features for the predictive model

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



REST API

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

Node-RED

data ingestion

SNAP4CITY

REST API

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

OOPS!

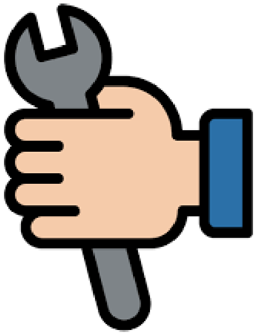
R

daily?

1																
2		dati ci sono ma non intero mese														
3		dato quasi mese completo ma c'è un buco														
4		dati ci sono intero mese														
5																
6		Gen2019	Feb2019	Mar2019	Apr2019	Mag2019	Giu2019	Lug2019	Ago2019	Set2019	Ott2019	Nov2019	Dic2019	Gen2020	Feb2020	Mar2020
7	PISA	15	0	20	30	31	30	16	0	0	0	0	19	28	29	31
8	SIENA	15	0	3	0	0	6	22	31	30	31	30	30	28	29	31
9																

Data Imputation

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



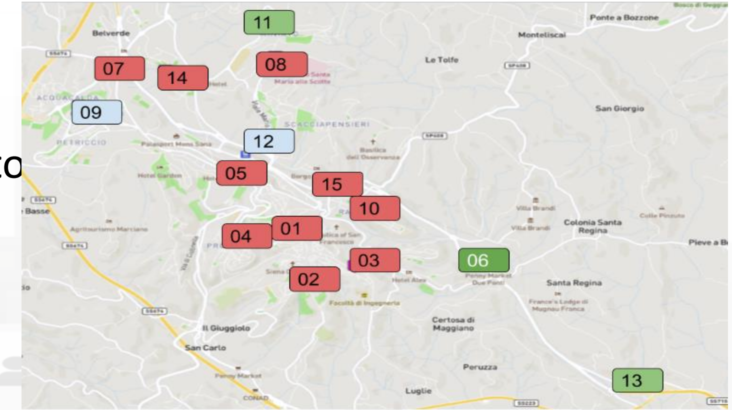
Data Imputation Strategies:

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

Clustering

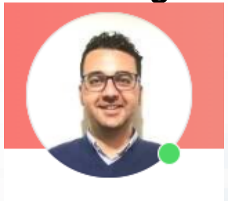
- A **clustering** approach has been applied in order to classify Pisa and Siena stations based on their mean trend H24 of bikes availability
  - This is also correlated to the typical services in the neighborhoods

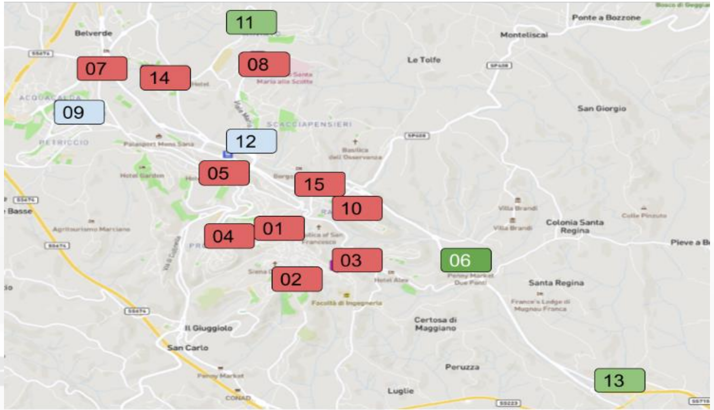
- **K-means** clustering method has been applied to identify clusters
  - The optimal number of clusters resulted to be equal to **3**, and it has been identified by using the **Elbow criteria**



Clustering

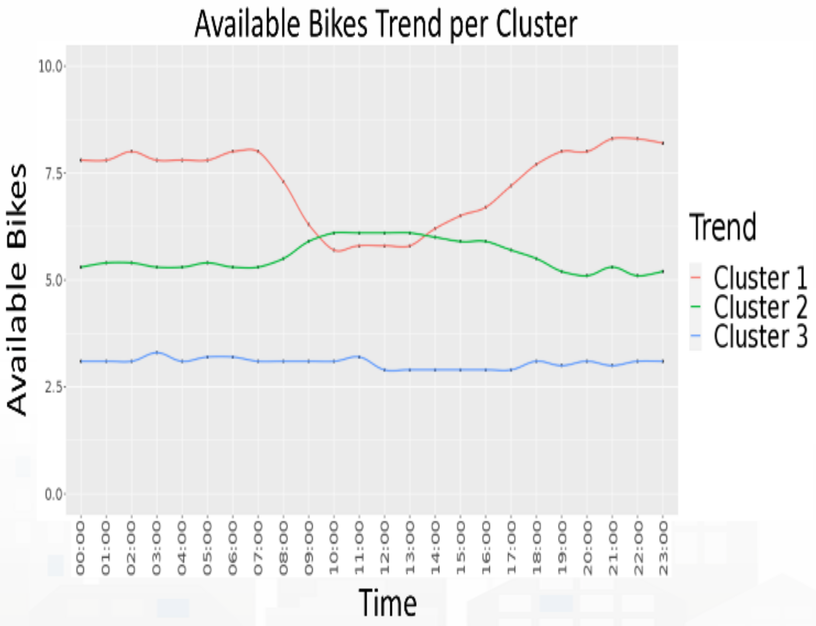
- Descriptive Statistics
- Trend Plots Analysis
- **Clustering**





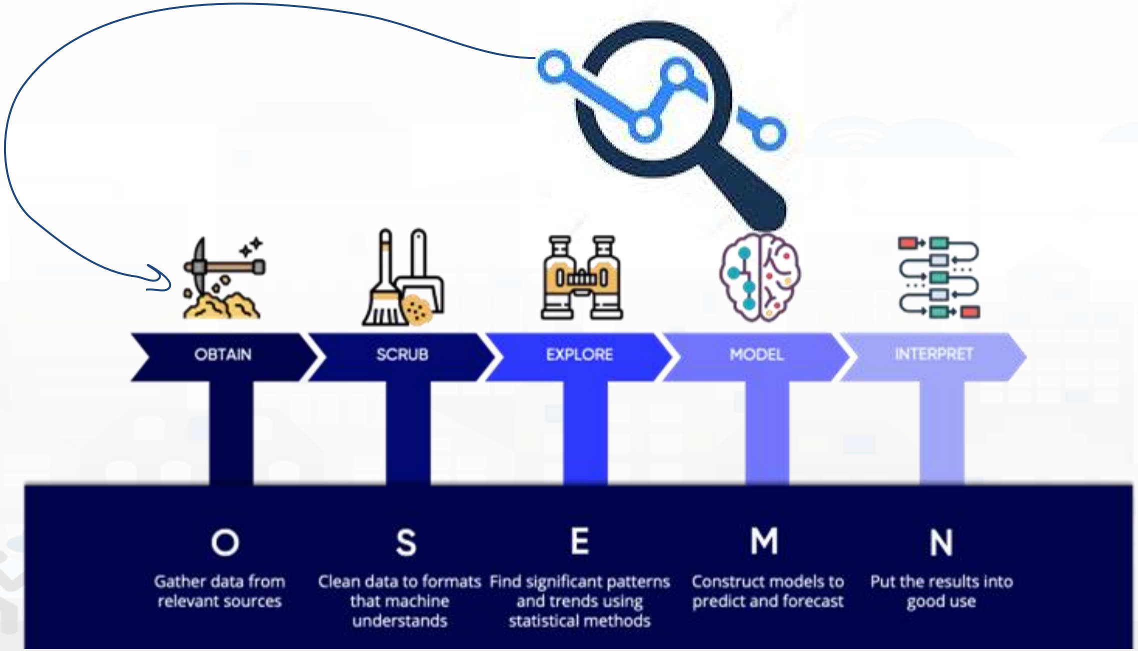
Clustering

- **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 & **data sources** 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

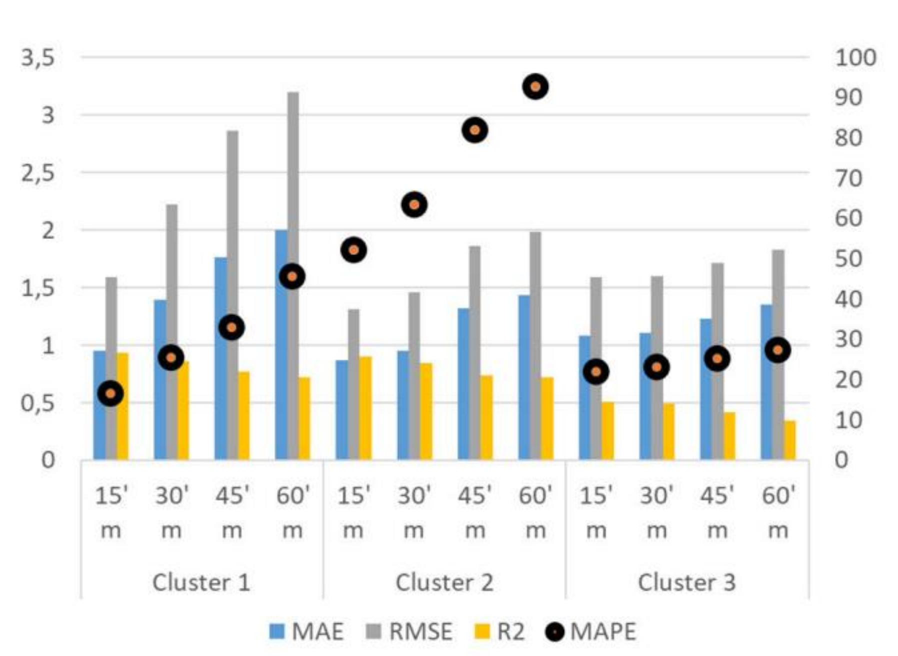
In general, Deep Recurrent Neural Networks architectures outperformed the ensemble learning techniques.

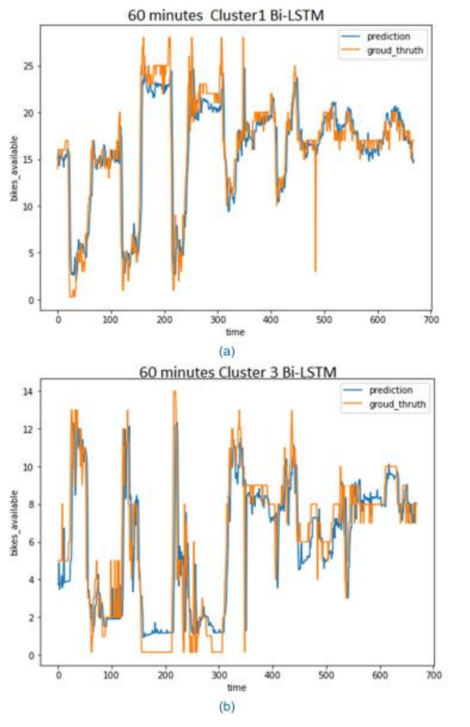
Overall, the best machine learning technique for the prediction of the number of available bikes turned out to be the Bi-LSTM.

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

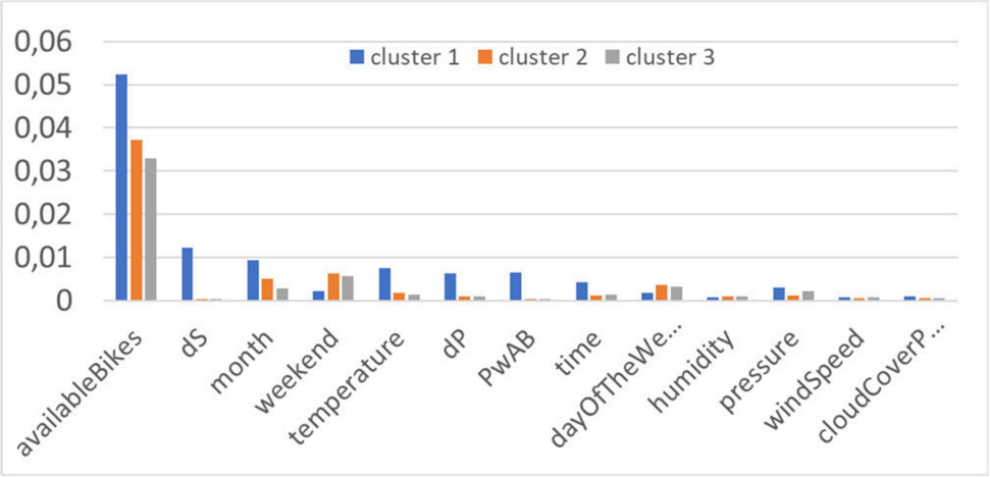




(a)
(b)

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



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.

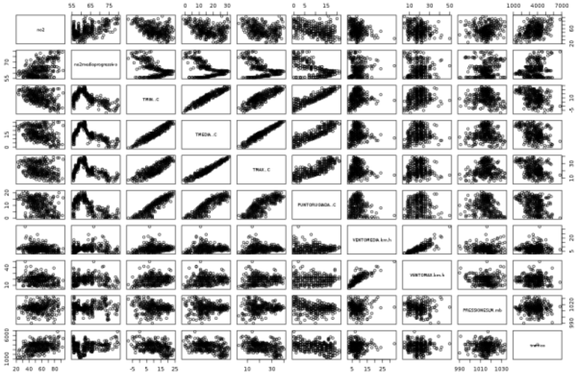
There are plenty of possibilities to conduct a feature selection analysis

- Linear Correlation Analysis
- Principal Component Analysis

Linear Correlation Analysis

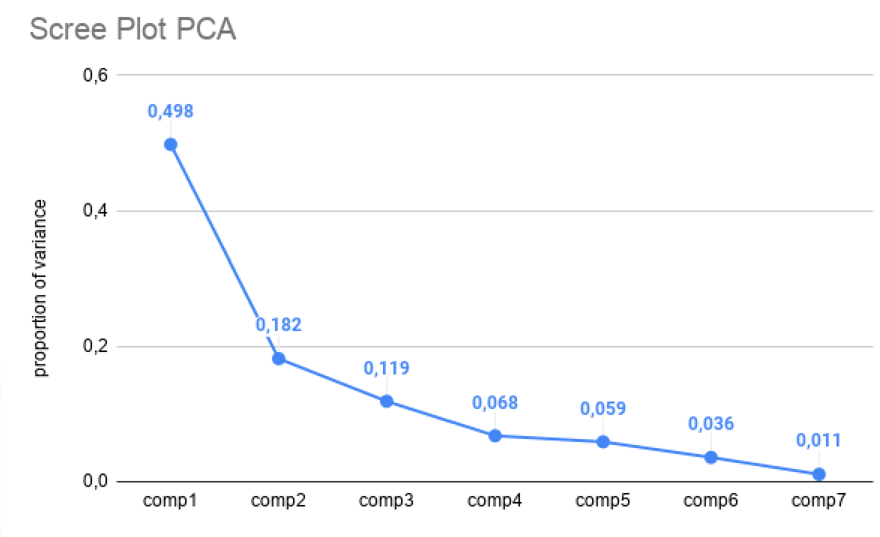
- Through correlation, we can predict one variable from the other.
- The logic behind using correlation for feature selection is that the good variables are highly correlated with the target.
- Otherwise If two variables are correlated, we can predict one from the other. Therefore, if two features are 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

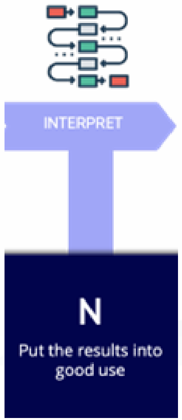
- PCA is a multivariate data analysis based on projection methods that results in a matrix that summarizes how our variables all relate to one another in different **principal components**.
- data reduction technique that transform the dataset into a compressed form that capture maximum information (**proportion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

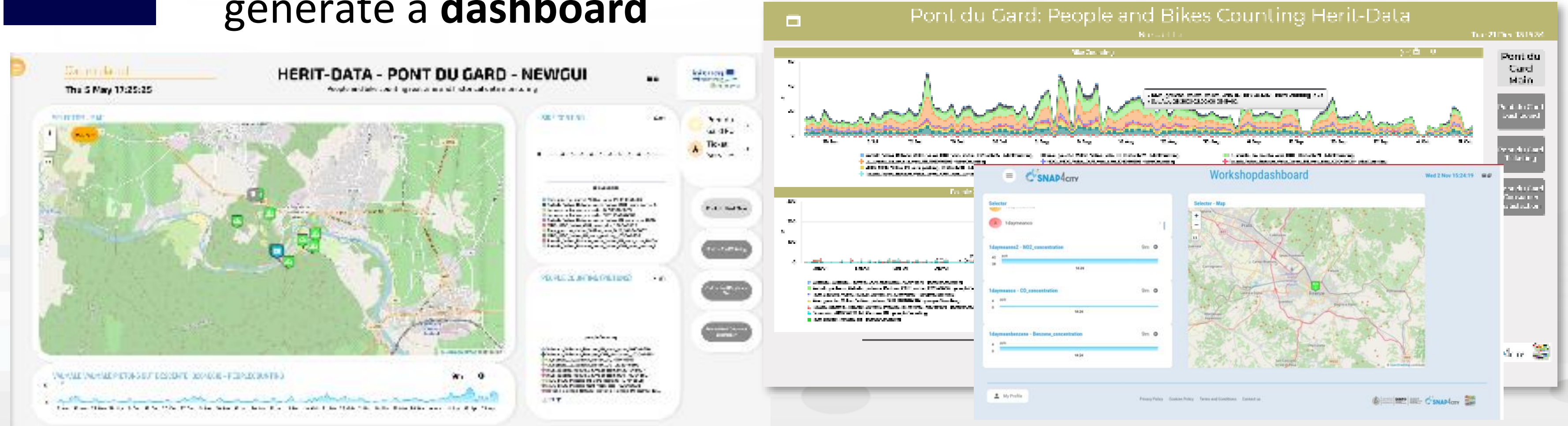
Table 2. Principal Components analysis (a part).

Time Series in good use- Monitoring Dashboard

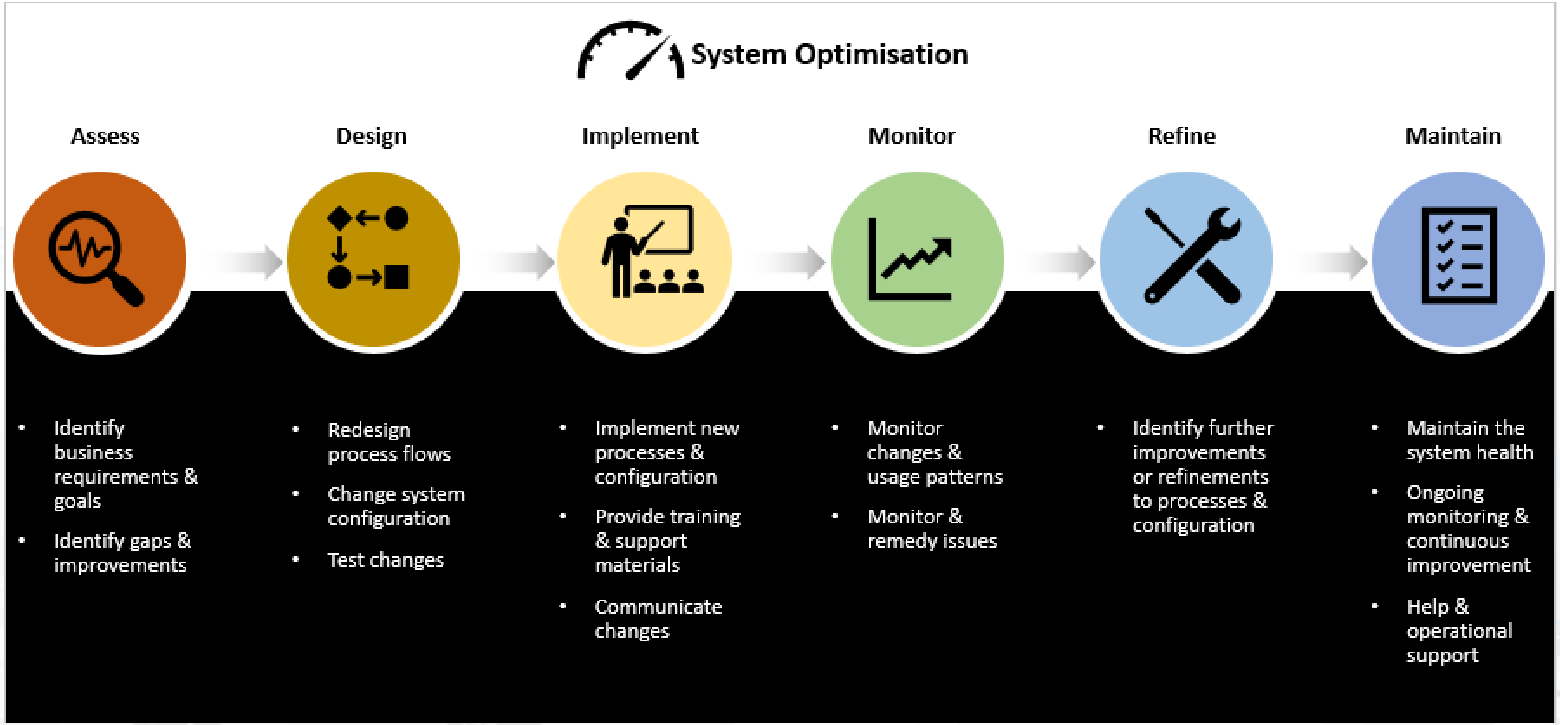


- The models developed can be inserted in **an automated process** that every day generates the input for the models and makes the predictions using a service of the SNAP4City platform, the IoTApp.
- The results are saved in the infrastructure and used to generate a **dashboard**

Node-RED



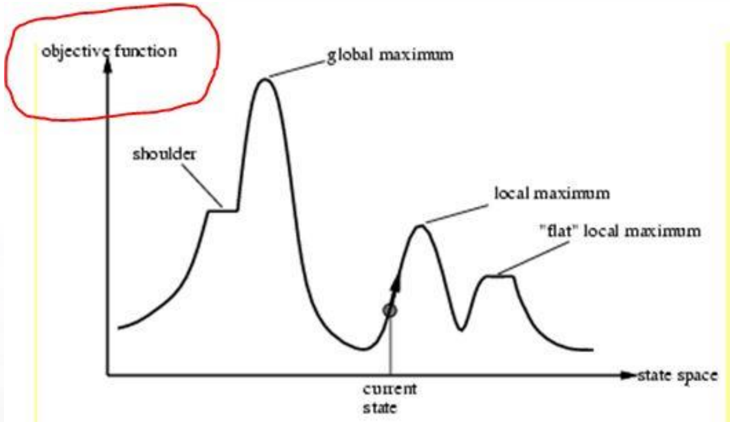
Optimization Systems



Optimization Methods: Introduction

What is Optimization?

- The process of finding the best solution to a problem within given constraints.
- Example: Finding the shortest path for a delivery truck to minimize fuel costs.



Why is it Important?

- Optimization helps in improving performance, reducing costs, and maximizing results.
- Used in various industries: manufacturing, logistics, AI, and economics.

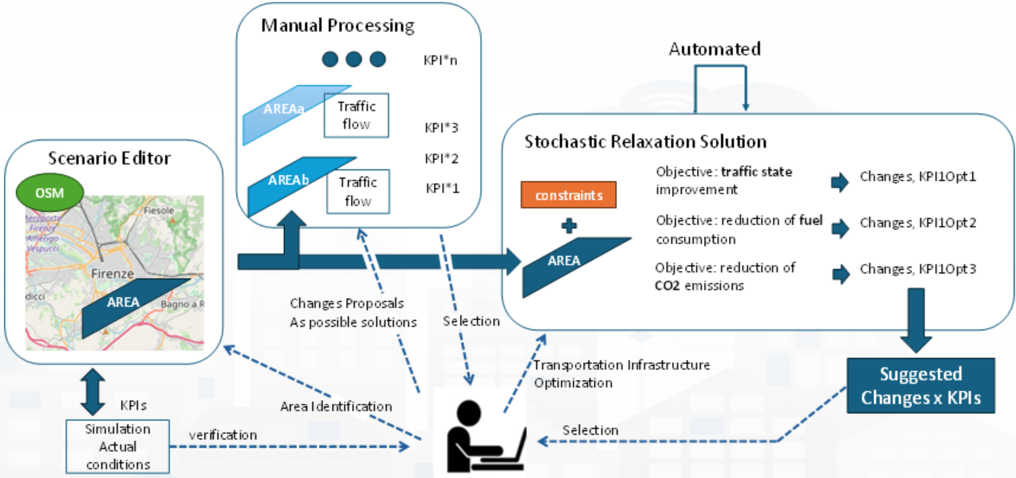
Optimization Methods for Function Minimization/Maximization

What Does Minimization/Maximization Mean?

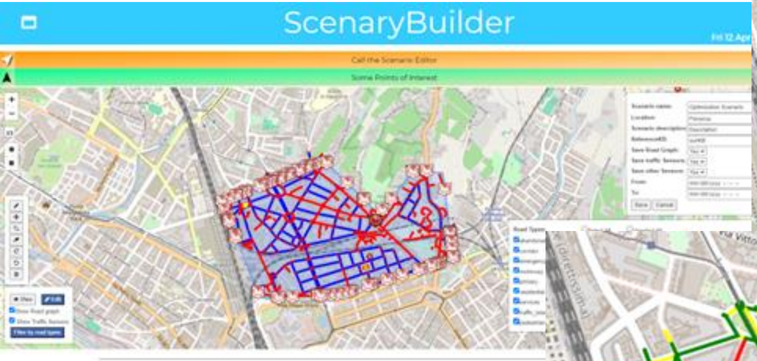
- In optimization, we often seek to minimize or maximize a function, such as cost, time, or energy, subject to constraints.
- Common in real-world problems like minimizing costs or maximizing efficiency.

Challenges:

- The problem may have multiple local minima/maxima.
- It's not always easy to find the global minimum/maximum, especially in complex or large search spaces.



Scenario



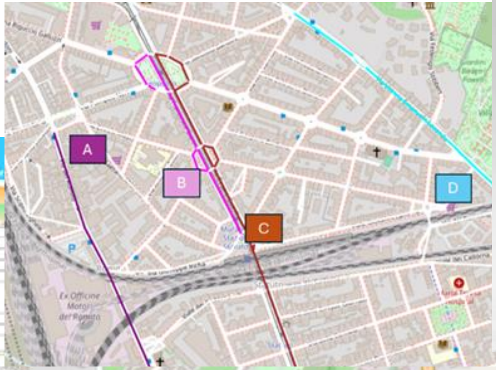


TABLE I. KPI OBJECTIVE FUNCTIONS IN THE INITIAL SCENARIO

	Traffic State KPI	*Fuel KPI*	*CO2 KPI*
Initial Scenario	115.475	25.680	165822

TABLE II. TRAVEL TIMES ON MAIN PATHS IN INITIAL SCENARIO

	Travel Times in seconds				
	Path A	*Path B*	*Path C*	*Path D*	*Total Time*
Initial Scenario	183.2	59.6	80.9	132.5	456.4

How to improve mobility

The possible changes on the road graph for generating a new configuration of transportation infrastructure are mainly on changing the **lane allocation per road direction**.

The computation of all possible combinations of road graph configurations is computationally infeasible. The scenario under analysis present 549 road segments. Not all of them can be changed for some reason according to the operator. The reasons to mark some road as blocked can be various: main inflow roads of the scenario at the boundaries, segments in which the traffic flow is injected, main directions, recently changed, not relevant, presence of control points, etc. Therefore, in the scenario the total amount of remaining road segments that can be changed in the case study was 216.

Given a road segment *s* its number of lanes considering both directions is *s.total_lanes*. The total number of possible combinations for the directions of a single road with multiple lanes are 2^total_lanes. If all the 216 roads are single lane a total number of combinations of , resulting in about 10^65. With 280 total number of lanes (an average of 30% of roads with two lanes), the number of combinations is in the order of **10^84**…Not all the combinations of changes are feasible as above stated.

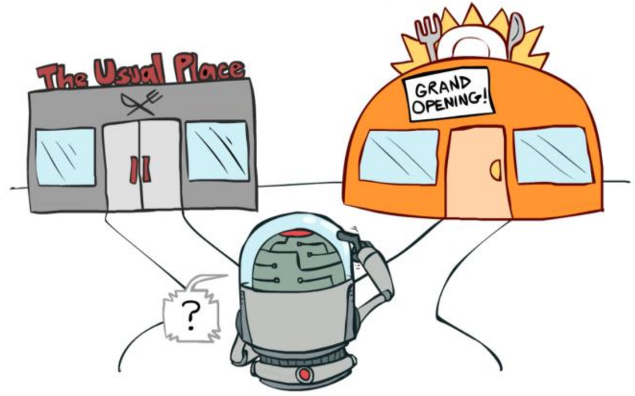
Metaheuristic Optimization Techniques

- Metaheuristic optimization techniques are a class of algorithms designed to find approximate solutions to complex optimization problems, particularly when the solution space is too large or difficult to explore exhaustively.

One of their most important characteristics is their ability to **balance exploration** and **exploitation**.

-Exploration refers to the process of searching new, unexplored areas of the solution space. In other words, it's about trying out solutions that are distant from current or previous solutions in order to uncover potentially better regions.

-Exploitation involves focusing on and improving regions of the solution space that are already known to be promising. It seeks to refine and optimize solutions that have already been identified as potentially good.



Method 1: Simulated Annealing (SA)

What is Simulated Annealing?

- A probabilistic technique inspired by the process of annealing in metallurgy, where controlled cooling is used to minimize energy states in a material.
- It helps escape local minima by allowing the system to explore higher-energy (worse) solutions at first and gradually "cooling" to find a good solution.

How it Works:

- Starts with a random solution.
- Iteratively explores neighboring solutions.
- Accepts a worse solution with a certain probability (controlled by a "temperature" parameter).
- As the temperature decreases, the algorithm becomes more likely to accept only better solutions.

Method 2: Tabu Search

What is Tabu Search?

- A local search algorithm that avoids cycling back to previously visited solutions (hence, the term "tabu") by maintaining a memory of previous states.
- It helps in overcoming local minima by intelligently choosing the next solution to explore, even if it's worse than the current one.

How it Works:

- Start with an initial solution.
- Iteratively move to neighboring solutions, but avoid revisiting recent ones by using a "tabu list."
- The search continues even if the current solution is not the best, but the algorithm is guided by the memory of past solutions.

Method 3: Genetic Algorithms

What is a Genetic Algorithm?

- A search heuristic inspired by the process of natural selection and genetics. It simulates evolution by using a population of potential solutions.
- The algorithm evolves solutions over time by selecting the best ones, applying crossover (mixing of solutions), and introducing mutations.

How it Works:

- Start with a population of random solutions.
- Select the best solutions based on a fitness function.
- Apply crossover (combining parts of two solutions) and mutation (random changes) to create new solutions.
- Repeat the process until a satisfactory solution is found.

Method 4: Particle Swarm Optimization (PSO)

What is Particle Swarm Optimization?

- A population-based optimization algorithm inspired by the social behavior of birds flocking or fish schooling.
- Solutions are treated as "particles" that move through the solution space, adjusting their positions based on their own experiences and the experiences of neighboring particles.

How it Works:

- Each particle adjusts its position based on two factors:
  - The best solution it has found (personal best).
  - The best solution found by its neighbors (global best).
- The algorithm balances exploration and exploitation of the search space to converge to an optimal solution.

Comparison of Optimization Methods

Simulated Annealing:

- Pros: Can escape local minima, good for complex, multi-modal functions.
- Cons: Can be slow and requires careful tuning of parameters.

Tabu Search:

- Pros: Efficient for large combinatorial problems, avoids cycling.
- Cons: Can get stuck in local optima if not well-tuned.

Genetic Algorithms:

- Pros: Good for highly complex problems, handles large search spaces.
- Cons: Can be computationally expensive, needs good parameter settings.

Particle Swarm Optimization:

- Pros: Simple, easy to implement, good convergence properties.
- Cons: May get stuck in local optima, requires parameter tuning.

https://colab.research.google.com/drive/1Q6bQdsyf4DldBHhgcSRam2JT6XLs-uVD?usp=sharing

Mobility Scenario Optimization Method Proposed

```
Parameters = Variable initialization // see in the following
CurrentBestScenario = {Load_Initial_Scenario()};
Solutions = [CurrentBestScenario]
Functional = "Traffic_State";
Iteration = 0
Annealings = {set_initial_value()}
Do:
    NewCases=New_Generation(CurrentBestScenario, Parameters
    For each C in NewCases
        TFR = Traffic_Flow_Reconstruction(C);
        If ( ErrControl(TFR) < ErrRefControl ) then
            KPI = KPI_Evaluation (TFR, Functional);
            Solutions.ProposeInTheBest (C, KPI);
        Endif
    Endfor

    If Annealings >0:
        // Select top 'Annealings' solutions
        AnnealingsSolutions = Solutions.GetTop(Annealings);

        For each A in AnnealingsSolutions
            NewCases = New_Generation (A, Parameters)
            For each C in NewCases
                TFR = Traffic_Flow_Reconstruction(C);
                If ( ErrControl(TFR) < ErrRefControl ) then
                    KPI = KPI_Evaluation (TFR, Functional);
                    Solutions.ProposeInTheBest (C, KPI);
                Endif
            Endfor
        Endfor

    CurrentBestScenario = Solutions.GetTop(N); //the best set
    Annealings = max(Annealings - 1, 0);
    Iteration ++;
While Max_Iterations>Iteration and not Early_Stop(KPI)
ResultCase = CurrentBestScenario
```

Figure 5 – Pseudocode of the proposed Stochastic Relaxation.

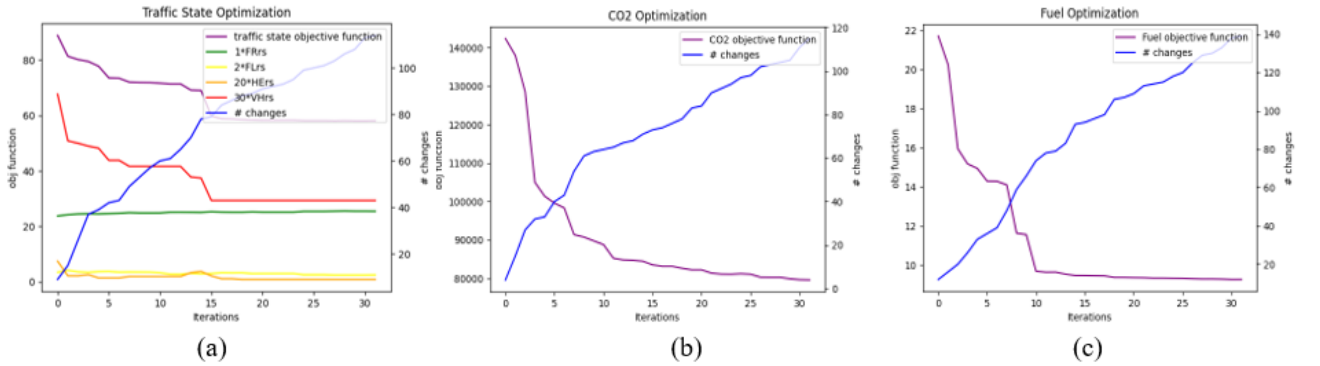


Figure 7 – Trends of the optimization process and the number of changes as function of the number of iterations regarding KPI as (a) traffic state, (b) fuel, and (c) CO2. (*Case NoLimits on Changes*).

Mobility Scenario Optimization Method Proposed

TABLE I. KPI OBJECTIVE FUNCTIONS IN THE INITIAL SCENARIO

	Traffic State KPI	*Fuel KPI*	*CO2 KPI*
Initial Scenario	115.475	25.680	165822

TABLE II. TRAVEL TIMES ON MAIN PATHS IN INITIAL SCENARIO

	Travel Times in seconds				
	Path A	*Path B*	*Path C*	*Path D*	*Total Time*
Initial Scenario	183.2	59.6	80.9	132.5	456.4

TABLE III. OPTIMIZATION RESULTS COMPARISON (*CASE NOLIMITS ON CHANGES)*

Case No limits	*KPI estimation on the best solution*		
Optimization Target	*Traffic State*	*Fuel*	*CO2*
Optim 1 Traffic State	**58.185**	9.432	86195
Optim 2 Fuel	59.844	**9.275**	80176
Optim 3 CO2	58.691	10.043	**79564**

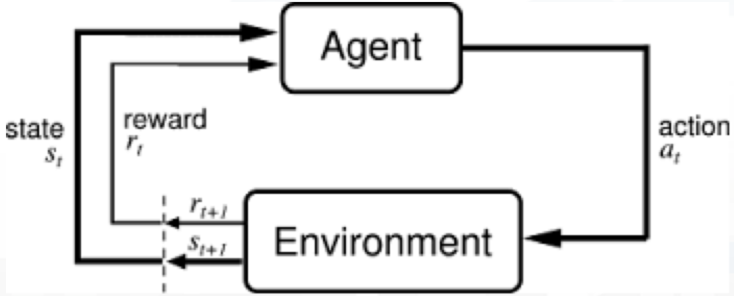
TABLE IV. MAIN ROADS TRAVEL TIME COMPARISON (CASE NOLIMITS ON CHANGES)

Travel Time [s]	*Path A*	*Path B*	*Path C*	*Path D*	*Total Time*
Initial Scenario	183.2	59.6	80.9	132.5	456.4
Optim 1 Traffic State	**87.9**	67.3	**48.4**	**70.1**	**273.8**
Optim 2 Fuel	93.4	64.7	70.1	87.6	316.0
Optim 3 CO2	90.4	**52.5**	52.8	78.5	274.3

Are there better solutions ?

Deep Reinforcement Learning (Deep RL) for Optimization

- Deep Reinforcement Learning combines **Reinforcement Learning (RL)** with **Deep Learning** techniques. The agent learns to make decisions by interacting with an environment, using neural networks to approximate the expected utility of a given action.



PROS

The agent "explores" different solutions and "exploits" the best-performing ones based on rewards.

CONS

its effectiveness depends on the problem size and the ability to learn effectively from the <u>environment</u>

Temporal-Differences Deep RL

Q-learning objective is to find an optimal policy regarding a sequential decision structured problem in which an agent needs to decide the action that maximizes the cumulative reward.

The main components of the system are:

State: the current configuration (ex. the city sequence for the TSP)

Action: the changes that the agent can perform (change the order of 2 cities to visit in the path)

Reward: the environment feedback (KPI length of agent path)

Q-function: Estimation of the quality of a couple state-action.

Policy: Mix of Exploitation (select best estimated action on the specific state) and Exploration (random action for exploring new possibilities)

Q-Learning with DNN for TSP

DNN can be used for Q-function approximation:

State: The current path that can be used as **input** for the DNN

Action: Can be the **output** from the DNN.

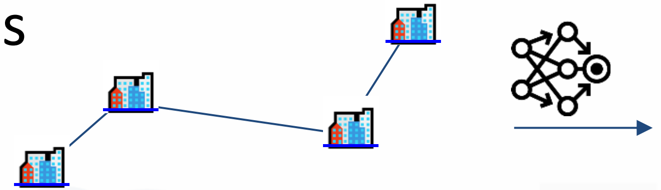
Functioning: For very possible change between two cities in the path the DNN will output the Q- values.

$$r = \frac{KPI(s) - KPI(s')}{KPI(s)}$$

$$Q(s,a) = r + \gamma \cdot \max_{a'} Q(s', a')$$

DNN Q-Learning workflow

s



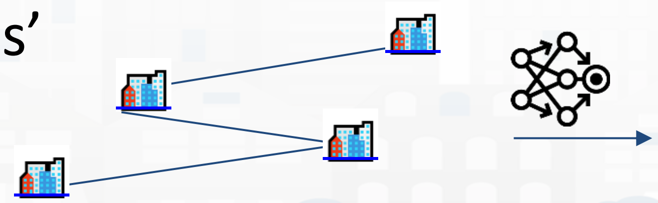
KPI(s)

policy π -> action a

s,a -> s'

Temporal Difference methodologies
use estimates of actions incrementally, updating the value future estimates of states and actions … next slide

s'



KPI(s')

$$r = \frac{KPI(s) - KPI(s')}{KPI(s)}$$

DNN Q-Learning workflow

```python
# Generate all possible pairs of cities to swap
swap_actions = list(itertools.combinations(cities, 2))
```

KPI(s_a1)
KPI(s_a2)
…
KPI(s_aN)

} max $Q(s,a) = r + \gamma \cdot \max_{a'} Q(s',a')$

Discount factor (γ): It determines how much future rewards influence the value of the Q-function. A higher γ (closer to 1) means that the agent places more importance on future rewards, while a lower γ (closer to 0) means that the agent prioritizes immediate rewards.

https://colab.research.google.com/drive/1Q6bQdsyf4DldBHhgcSRam2JT6XLs-uVD?usp=sharing

Final results from the optimization Colab of this lesson