Shared-bike services: from open data platforms to a dataviz application

FOSS4G2019 - Bucharest

Raphaël Delhome(*), Damien Garaud

Introduction

Shared-bike services

Velo'v (Lyon)

- Shared-bike rental service in large cities
- Small-duration rents
- Stations and availability

Major challenges

- Bike (resp. bike stand) availability ?
- Bike-sharing station classification ?
- Data pipeline design ?

Outline

(Part 1) Handle open geospatial data

(Part 2) Bike-sharing station unsupervised classification

(Part 3) Bike and station short-term availability prediction

(Part 4) Web application demo

Data overview

Open geospatial data

- Lyon, France

- Bordeaux, France

Which data?

id	bs	abs	ab	status	last_timestamp
10063	34	10	23	OPEN	2017-07-08 23:49:09
10021	19	0	0	CLOSED	2017-07-08 00:30:12
8038	20	6	14	OPEN	2017-07-08 23:49:26
7045	20	13	7	OPEN	2017-07-08 23:52:43

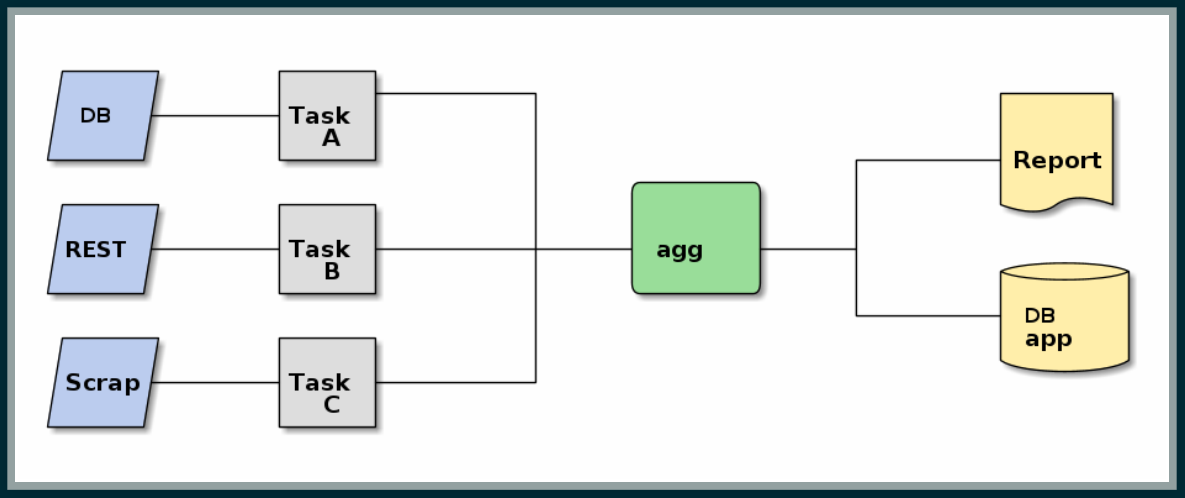
General Bikeshare Feed Specification

https://github.com/NABSA/gbfs

File Name	Required	Defines
gbfs.json	Optional	Auto-discovery file that links to all of the other files published by the system. This file is optional, but highly recommended.
system_information.json	Yes	Describes the system including System operator, System location, year implemented, URLs, contact info, time zone
station_information.json	Conditionally required	Mostly static list of all stations, their capacities and locations. Required of systems utilizing docks.
station_status.json	Conditionally required	Number of available bikes and docks at each station and station availability. Required of systems utilizing docks.
free_bike_status.json	Conditionally required	Describes bikes that are available for rent. Required of systems that don't utilize docks or offer bikes for rent outside of stations.
system_hours.json	Optional	Describes the hours of operation for the system
system_calendar.json	Optional	Describes the days of operation for the system
system_regions.json	Optional	Describes the regions the system is broken up into
system_pricing_plans.json	Optional	Describes the system pricing
system_alerts.json	Optional	Describes current system alerts

Data pipeline

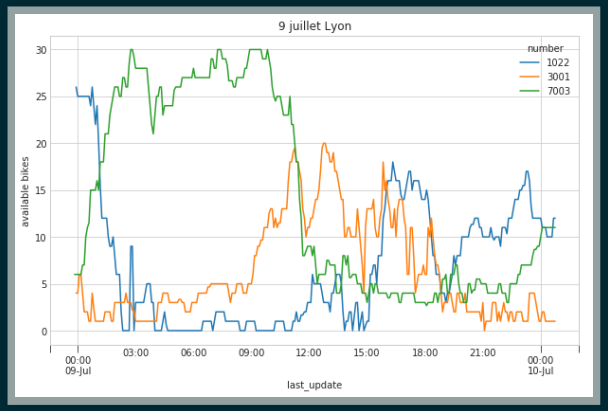
- Build a Python data pipeline thanks to Luigi
- Get, transform and store the data
  - get data every five minutes (`json`, `shp`)
  - in-base storage (`PostgreSQL+Postgis`)
  - feature engineering and ML treatments



Bike-sharing station classification

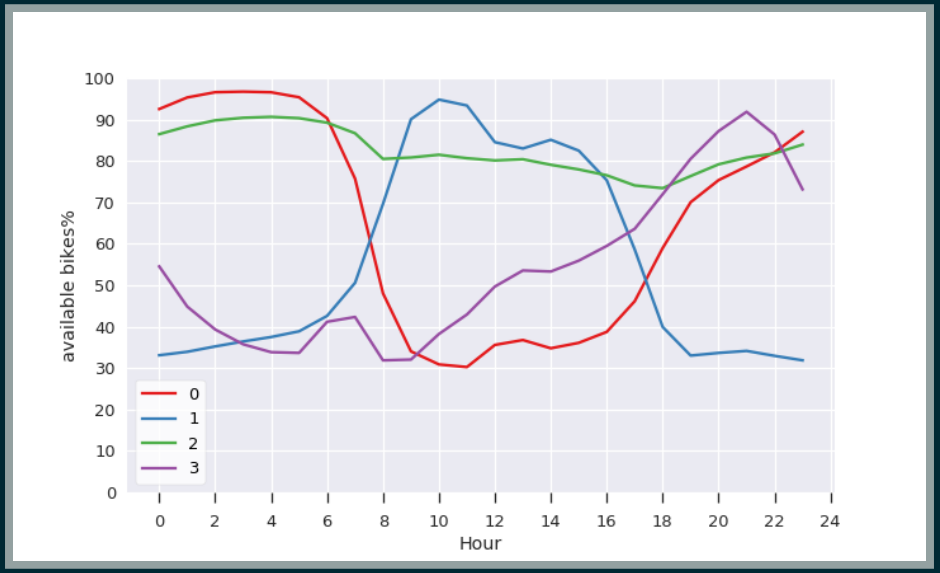
Objective

- Classify bike-sharing station according to their usage by customers
- *Main idea* = group stations that look similar
- *... What does it mean?* => Focus on the time series



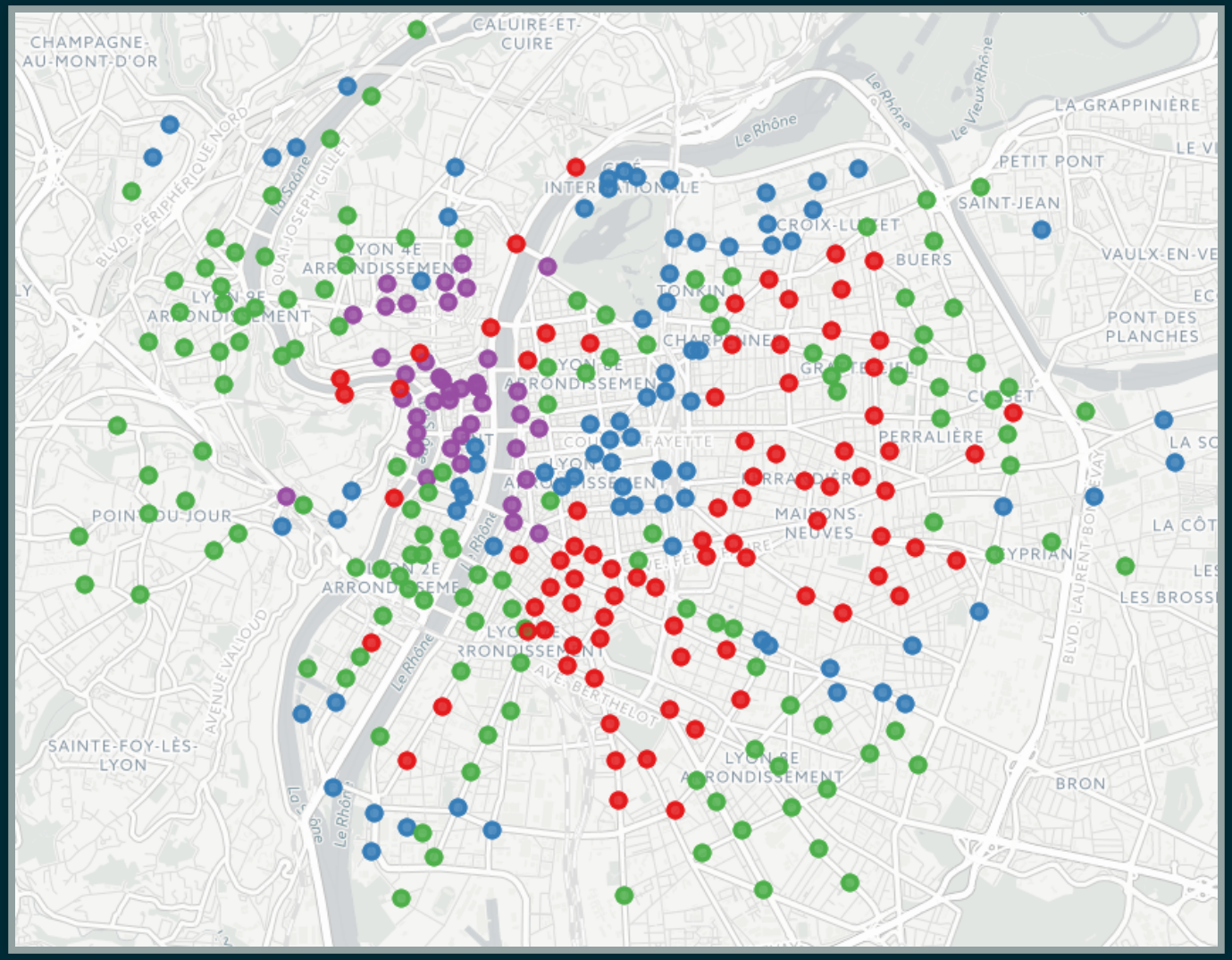
K-means clustering

- Inspired from a similar work of James Lawlor
- One profile = one individual
- Group similar individuals together
- Deduce stations profiles



ex: 4 clusters in Lyon

Clustered station mapping



Shared-bike availability prediction

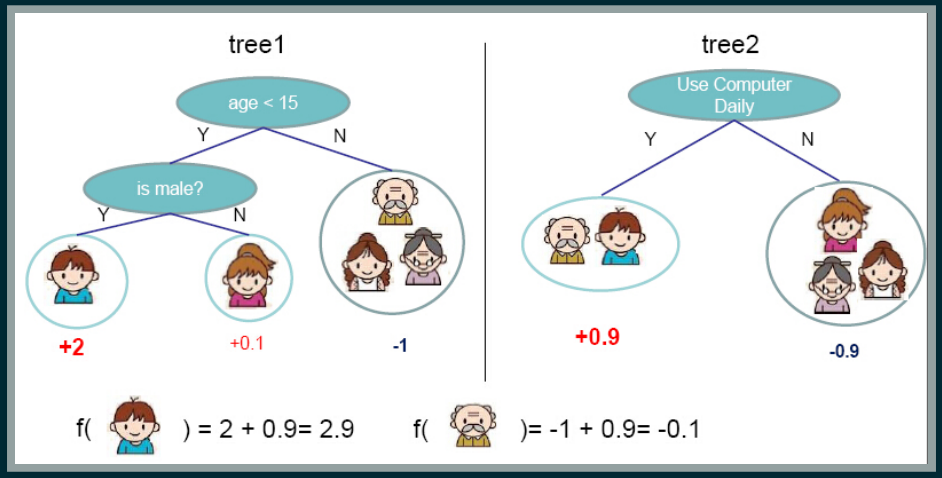
Objective

- Know if some bikes (*resp*. stations) will be available in the next few minutes
- *Main idea* = Predict **future** availability with availability **history**
- ... *What does it mean?* => Supervised learning to learn an availability probability

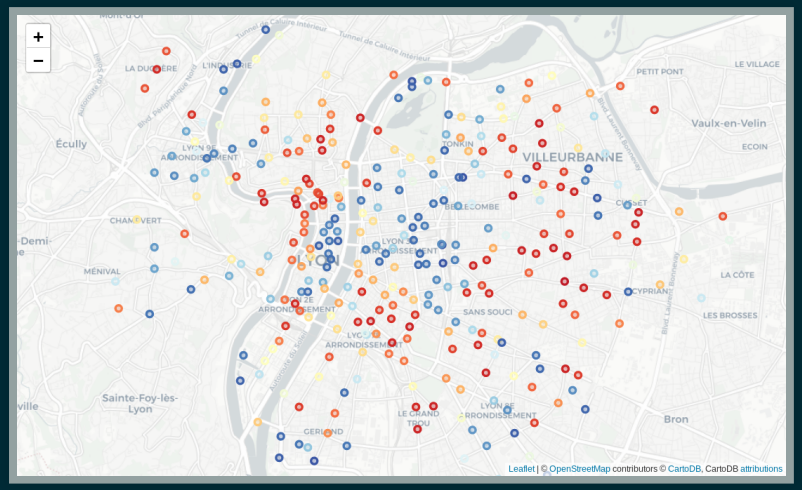
XGBoost method

Use a boosting tree method :

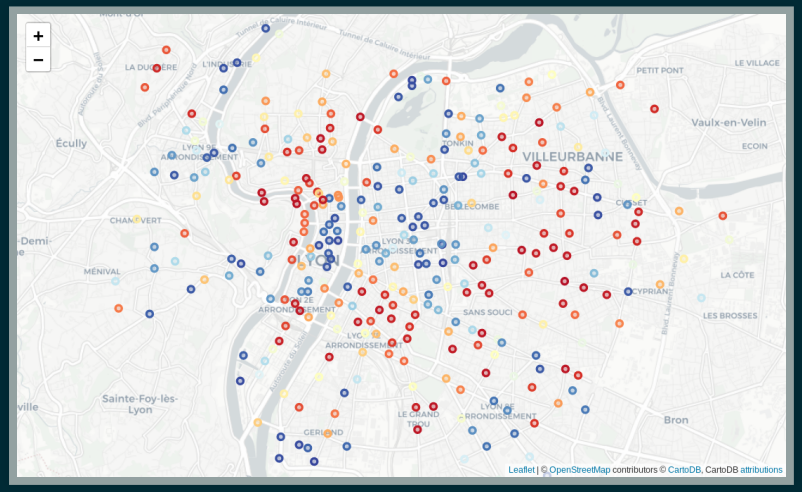
- to predict **Y** (availability probability at H+1)
- starting from **X** (hour, day, available bikes at H, ...)



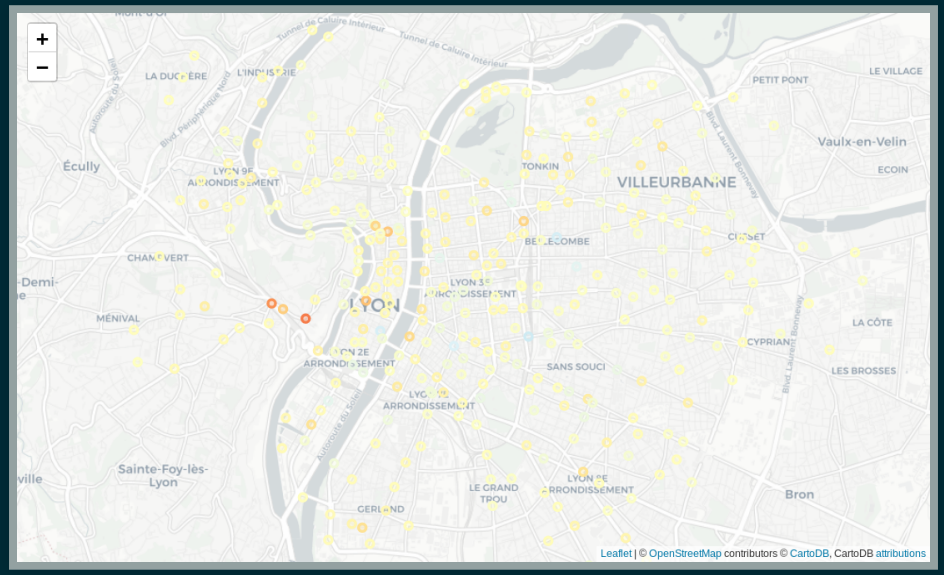
Results



(^) Prediction; Ground-truth (v)



Without tuning features,
RMSE = 0.095



(^) Error

Demo

Web application: http://data.oslandia.io/bikes/

Conclusion

Conclusion and perspectives

- Addressing some simple research questions with some open geospatial dataset
- From data source to database (ETL-like) with Luigi
- Production of an API to visualize data => towards production?
- Online learning: keep on gathering data, and learn continuously

Thanks for your attention!

Questions?

damien.garaud@oslandia.com
raphael.delhome@oslandia.com

See more on Oslandia's blog and on
github.com/garaud/jitenshea