P3_report

December 8, 2016

P3: Wrangle OpenStreetMap Data
Bordeaux (France)
Julien Delbove

1 Introduction

This project has been realized with the OpenStreetMap data of the city of **Bordeaux**. Bordeaux is a well known city of France because of its wine. Futhermore, it is the closest city from where I live available in the preselected metro area from Map Zen.

The links to the dataset are: * Preselected metro area from Map Zen: https://mapzen.com/data/metro-extracts/metro/bordeaux_france/ * Direct link to the data: https://s3.amazonaws.com/metro-extracts.mapzen.com/bordeaux_france.osm.bz2

The initial data exploration has been made on a reduce sample of the dataset. One element over 50 were kept in order to a reasonable sample size.

2 Problems Encountered in the Map

After an initial exploration of the data by parsing the file several times with the aims to look at different information, I created a function which aims at converting the dataset into a json structure. In addition, this function take as input an audit function that could to used to: * displaying some informations necessary for the audit of some elements * correcting the elements that have been flagged as wrong

```
In [2]: def convert_in_json(filename,audit_function):
```

The exploration of the data showed the following issues:
1. In the housenumber field, several inconsistencies have been dedected: name instead of numbers (e.g.: 'Mama Shelter'), single letters instead of numbers (e.g.: 'G'), non uniform convention of numbering that looks correct ('31bis' vs '26 Ter'), postcode instead of housenumber (e.g.: '33270') and some others on which it is difficult to conclude (e.g.: '38/40')
2. In the street name, the following issues have been dedected: harmonization of the street type capital letters, no street type (e.g.: Docteur Fauché), housenumber at the beginning of the street (e.g.: 68 Avenue Jean Jaurès)
3. In the postcode, an error has been found for one case: '33185 LE HAILLAN'
4. The same city names are sometimes written in different way: full capital letters of not (e.g.:

BORDEAUX vs Bordeaux), since the dataset is dealing with french name there is also the problem of the accent (e.g.: Saint-Médard en Jalles vs saint medard en jalles) and sometimes the use of dash for compound names (e.g.: Saint-Médard-en-Jalles)

2.1 Housenumber field

This issue looks very difficult to solve without additional information. At this stage, I know that the information is probably wrong but I don't see any way to clear the issue of having a wrong numbering. To solve this kind of issue programatically, the solution would be to get information from a third party (either another database or having the possibility to request the information to somebody)

2.2 Street name

I updated the street name based on the following algorithm: * check that the first word of the street name is a number (if yes and there is no housenumber then I updated the housenumber) * check the streetname is in a dictionnary of correction to make to the field (if yes, streetname is updated)

2.3 Postcode

The complete list of postcode is the following:

```
In [6]: a=['33490', '33850', '33240', '33360', '33491', '33200', '33320',
           '33710', '33612', '33610', '33185', '33402', '33185 LE HAILLAN',
           '33910', '33290', '33110', '33070', '33500', '33420',
           '33130', '33230', '33640', '33760', '33700', '33310', '33170',
           '33150', '33400', '33330', '33530',
           '33525', '33370', '33480', '33550', '33210', '33440', '33270',
           '33460', '33720', '33600', '33670',
           '33800', '33077', '33100', '33160', '33380', '33138', '33126',
           '33127', '33300', '33650', '33450', '33410', '33000', '33140',
           '33750']
```

The node that had the postcode '33185 LE HAILLAN' had not city name. Therefore the postcode has been updated as well as the city name.

2.4 City name

For the city name, I made a dictionnary of corrections. Each time the city name was found in the dictionnary, it was update by the "correct" value. I made the choice of city name with Capital letter only at the beginning of the word and to keep the accent and dash when they exist. To be more internationaly readable, the choice to remove all accents may be better.

3 Data Overview

The size of the sample file with one elements every 50 is: $22Mo$
The size of the file with which I created the mongoDB database is: $110Mo$

Once the mongo BD database created, I made the following MongoDB queries to get the following statistics:

number of unique users

Number of unique users 935
Most active users: [{u'count': 897295, u'_id': None}, {u'count': 70038, u'_id': u'mides'}, {u'count': 58998, u'_id': u'jfnif'}, {u'count': 56185, u'_id': u'guiguid'}, {u'count': 30119, u'_id': u'St\xe9phane Lecorn\xe9'}]

```
In [ ]: #number of unique users
        pipeline1 = [
                {"$group":{"_id":"$created.user","count":{"$sum":1}}},
                {"$sort":{"count":-1}},
                #{"$limit":10},
                ]

        result = [doc for doc in db.osm.aggregate(pipeline1)]
        print "Number of unique users", len(result)
        print "Most active users:",result[0:5]
```

number of nodes and ways

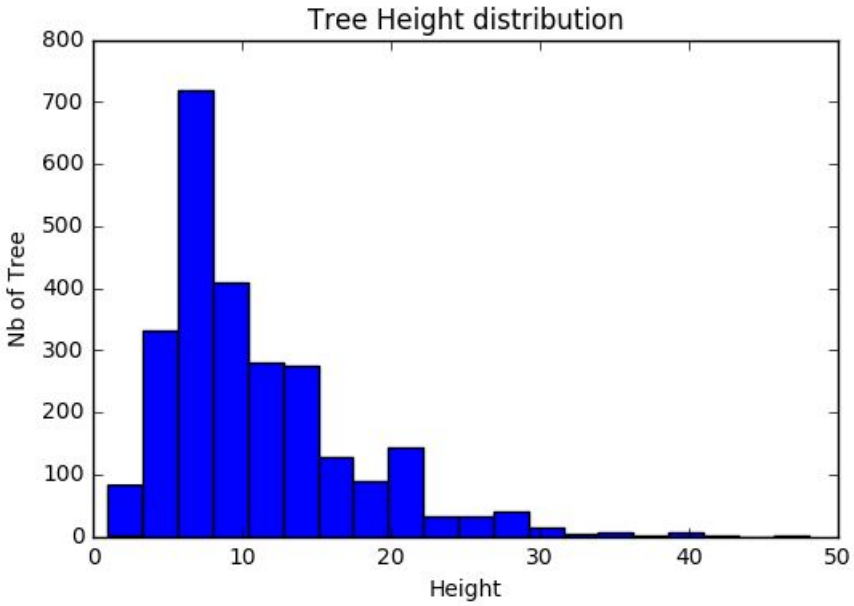
Nb of nodes: {u'count': 463555, u'_id': u'null'}
Nb of ways: {u'count': 80542, u'_id': u'null'}

```
In [ ]: #number of nodes
        pipeline2 = [
                {"$match":{"type":"node"}},
                {"$group":{"_id":"null","count":{"$sum":1}}},
                ]

        result = [doc for doc in db.osm.aggregate(pipeline2)]
        print "Nb of nodes:",result[0]


        #number of way
        pipeline3 = [
                {"$match":{"type":"way"}},
                {"$group":{"_id":"null","count":{"$sum":1}}},
                ]

        result = [doc for doc in db.osm.aggregate(pipeline3)]
        print "Nb of ways:",result[0]
```

number nodes representing trees, most popular tree species and distribution of tree heights

Nb of trees: {u'count': 2957, u'_id': u'null'}
Most numerous tree species: [{u'count': 512, u'_id': u'Platanus x hispanica'},

{u'count': 150, u'_id': u'Tilia tomentosa'}, {u'count': 93, u'_id': u'Fraxinus angustifolia'}, {u'count': 93, u'_id': u'Tilia cordata'}, {u'count': 92, u'_id': u'Quercus robur'}]
List of Tree Heights: [{u'_id': ObjectId('584730a71164031fbc73a05c'), u'height': u'6'}, {u'_id': ObjectId('584730a71164031fbc73a064'), u'height': u'9'}, {u'_id': ObjectId('584730a71164031fbc73a06c'), u'height': u'9'}, {u'_id': ObjectId('584730a71164031fbc73a074'), u'height': u'9'}, {u'_id': ObjectId('584730a71164031fbc73a07c'), u'height': u'8'}]



Distribution of tree heights:

```
In [ ]: #number of trees
        pipeline4 = [
                {"$match":{"type":"node","natural":"tree"}},
                {"$group":{"_id":"null","count":{"$sum":1}}},
                ]

        result = [doc for doc in db.osm.aggregate(pipeline4)]
        print "Nb of trees:",result[0]


        #most popular species among Bordeaux tree
        pipeline5 = [
                {"$match":{"type":"node","natural":"tree",
                           "species":{"$exists":"true"}}},
                {"$group":{"_id":"$species","count":{"$sum":1}}},
                {"$sort":{"count":-1}},
                {"$limit":10},
                ]

        result = [doc for doc in db.osm.aggregate(pipeline5)]
        print "Most numerous tree species:",result[0:5]
```

```
#tree heights
pipeline6 = [
        {"$match":{"type":"node","natural":"tree",
                   "height":{"$exists":"true"}}},
        {"$project":{"height":1}},
        ]

result = [doc for doc in db.osm.aggregate(pipeline6)]
print "List of Tree Heights:",result[0:5]
```

4 Additional Ideas

4.1 Ideas for improvement

I see two ways to improve the data:

- From the various information I read concerning the openstreetmap format, I have the impression that the standard is a bit desorganized. Let's take as an example the tree description. Tree are defined with node elements and specific attribute have been define to enter the different tree characteristic. A a consequences, if you don't exactly what attribute value you have to look after to get information on tree, you are lost. Trees are defined in the node with the attribute "natural=tree". At this stage, I would put in place a convention like for the address. All attributes relative to the trees need to start with "tree:" (e.g. tree:species, tree::height. . . .).

- A second way to improve the data quality would be from my opiniontо provide an smartphone application where people would get access to a user interface that would help them to customize how their house, prefered amenities, work or leisure place are entered in the osm database (GPS of the smartphone could be used to create nodes for elaborated shape. In addition the application could also ofer some services like guidance system and if you ask for an address that has been customize the service will display some content proposed by the person that has updated it (advertissement for restaurant, funny picture for leisure places, ...)

4.2 Benefits and problems with ideas for improvement

- The benefit of having a more structured database are that it is easy to fill (more understanble what each field means) and that it is easy to find/parse information since the structure is harmonized. The main issue is that this rigidity is at the detriment of the current flexibility of the OSM database. If a elment doen't fit to the structure, it can prevent somebody/ and organization to make the effort to input data in it. In addition, another problem would be the huge effort it would cost to make the transformation of all the already existing entries.

- The main benefit of app idea would be to able to ensure the data of the database are corrects: otherwise the other services would not work. The second benefit would be that it would be possible to propose a service that will help bussiness to get a nice customization if they what to be visible. The main problems are the cost of the application and bundle of services, legal

aspect in case some detrimental information are put for addresses because of grief and the penetration of such kind of application for smartphone user population.

5 Conclusion

As a conclusion to this study, I have been able to process an OSM dataset. I explored the dataset with functions I have created. I implemented a conversion function and audit functions in order to improve the quality of the addresses. I created a MongoDB with the improved dataset and I computed several statistic of the dataset. In addition, proposed 2 ideas to improve the quality of the current dataset.
The dataset for the aglomeration of Bordeaux looks pretty complete. A lot of information in addition to the address and map seem to be available, going from tree location and species, trafic signal, leisure places,... However, I think also that it would interesting to graphically display the information stored in the database to get a qualitative assessment of whether the information are globally correct or not.