

# Wrangle\_Openstreetmap\_Data

March 6, 2016

## 1 Wrangling Openstreetmap Data

### 1.1 1. Introduction

Choose any area of the world from <https://www.openstreetmap.org> , and download a XML OSM dataset. The dataset should be at least 50MB in size (uncompressed).

Use the Overpass API to download a custom square area. Explanation of the syntax can found in the [wiki](#) . In general you will want to use the following query:

```
(node(minimum_latitude, minimum_longitude, maximum_latitude, maximum_longitude);<);out meta;
```

e.g.

```
(node(51.249,7.148,51.251,7.152);<);out meta;
```

the meta option is included so the elements contain timestamp and user information. You can use the Open Street Map Export Tool to find the coordinates of your bounding box. Note: You will not be able to use the Export Tool to actually download the data, the area required for this project is too large.

#### 1.1.1 1.1 Selected map area

The selected area to work with is all the area near **Santander - Spain**, given by this bounding box query:

```
<osm-script>
  <union into="_">
    <bbox-query e="-3.7086" into="_" n="43.4988" s="43.3496" w="-4.0574"/>
    <recurse from="_" into="_" type="up"/>
  </union>
  <print e="" from="_" geometry="skeleton" limit="" mode="meta" n="" order="id" s="" w=""/>
</osm-script>
```

### 1.2 2. Problems encountered in the map

Once the data is exported and downloaded as a **OSM** file we can parse it. The idea is to transform the **XML** format of the input file to a **JSON** format, interpretable and ready to import into MongoDB.

This code could be find in [Wrangle\\_Openstreetmap\\_Data](#) repository from my github account.

Running the code selecting an **OSM** file as the input will result in a **JSON** file as output. Importing a JSON file into mongoDB is as easy as executing the command (note that you must have mongod installed):

```
mongoimport --db udacityP3 --collection near_santander --file near_santander.osm.json
```

The main problems found while processing the data are:

- Undefined keys
- is\_in field hasn't a consistent format

### 1.2.1 2.1 Undefined keys

Lets see al the different keys in the data. To do it, first we create a new collection in MongoDB with the keys as the 'id' (and with value the times they appear for future purposes).

```
db.near_santander.mapReduce(  
  function() {  
    for (var key in this) { emit(key, 1); }  
  },  
  function(key, n) {  
    return Array.sum(n);  
  },  
  {out: "santander_keys"}  
)
```

And querying the new collection:

```
db.santander_keys.distinct('_id')
```

we can see there's a key named 'undefined'.



[459]	twitter
[460]	type
[461]	undefined
[462]	usage
[463]	vehicle

Figure 1:

The way we fix this is ommit them in the map parse process.

<https://github.com/jmonterrubio/Wrangle-Openstreetmap-Data/blob/master/map-parser/map/tags.py#L15>

### 1.2.2 2.2 is\_in field hasn't a consistent format

Executing this query

```
db.near_santander.distinct('is_in',{is_in:{$exists:true}})
```

A sample screenshot:

we can see that 'is\_in' values have different formats. A list of things that may be fixed are: \* Use semi-colons to separate entities instead of commas \* Use same language (in this case for consistency, everything in spanish) \* Add well known entities (España, Europa) if missed. \* Remove duplicated entities \* Order the entities from smaller to bigger

Before trying to fix them, lets look for more information about the key: [http://wiki.openstreetmap.org/wiki/Key:is\\_in](http://wiki.openstreetmap.org/wiki/Key:is_in). We can see there isn't a mandatory way to fill the field, so lets implement the field fix.

In this case, the code implementing this fixes could be found at <https://github.com/jmonterrubio/Wrangle-Openstreetmap-Data/blob/master/map-parser/map/region.py>

## 1.3 3. Overview of the Data

This section contains basic statistics about the dataset and the MongoDB queries used to gather them.

- File sizes

[79]	Torrelavega; Cantabria; Cantabria; España; Europe
[80]	Soto de la Marina, Cantabria, España
[81]	Guarnizo; El Astillero; Cantabria; Cantabria; España; Europe
[82]	Loreda, Ribamontán al Mar, Cantabria, España
[83]	Muriedas; Camargo; Cantabria; Cantabria; España; Europe
[84]	Soto La Marina, Cantabria, España
[85]	Cantabria; España; Europe; Santander
[86]	Cantabria; España; Europe; Medio Cudeyo; San Salvador
[87]	Solares, Medio Cudeyo, Cantabria, Spain

Figure 2:

```
near_santander.osm ..... 76 MB
near_santander.osm.json .... 82 MB
```

- Number of documents

```
> db.near_santander.count()
354970
```

- Number of nodes

```
> db.near_santander.count({"type":"node"})
319053
```

- Number of ways

```
> db.near_santander.count({"type":"way"})
35874
```

- Number of unique users

```
> db.near_santander.distinct("created.user").length
251
```

- Top 1 contributing user

```
> db.near_santander.aggregate([
  {"$group":
    {"_id":"$created.user", "count":{"$sum":1}}
  },
  {"$sort":
    {"count":-1}
  },
  {"$limit":1}
])
/* 1 */
{
  "_id" : "Emilio Gomez",
  "count" : 164316
}
```

- Number of users appearing only once (having 1 post)

```

> db.near_santander.aggregate([
  {"$group":
    {"_id": "$created.user", "count": {"$sum": 1}}
  },
  {"$group":
    {"_id": "$count", "num_users": {"$sum": 1}}
  },
  {"$match":
    {"_id": 1}
  }
])
/* 1 */
{
  "_id" : 1,
  "num_users" : 62
}

```

- Number of different keys

```

> db.near_santander.mapReduce(
  function() {
    for (var key in this) { emit(key, 1); }
  },
  function(key, n) {
    return Array.sum(n);
  },
  {out: "near_santander_keys"}
)
> db.near_santander_keys.count()
479

```

- Top 10 used key

```

> db.near_santander_keys.find(
  {"_id":
    {"$nin: ["_id", "id", "created", "type"]}
  }).
  sort({"value": -1})
/* 1 */
{
  "_id" : "pos",
  "value" : 319056
}

/* 2 */
{
  "_id" : "node_refs",
  "value" : 35914
}

/* 3 */
{
  "_id" : "source",
  "value" : 30892
}

```

```

/* 4 */
{
  "_id" : "highway",
  "value" : 22134
}

/* 5 */
{
  "_id" : "name",
  "value" : 14008
}

/* 6 */
{
  "_id" : "building",
  "value" : 13552
}

/* 7 */
{
  "_id" : "created_by",
  "value" : 10932
}

/* 8 */
{
  "_id" : "source:date",
  "value" : 10891
}

/* 9 */
{
  "_id" : "building:levels",
  "value" : 8689
}

/* 10 */
{
  "_id" : "address",
  "value" : 8491
}

```

- Number of keys used only once

```

> db.near_santander_key.count({value:1})
111

```

- Top 10 amenities

```

> db.near_santander.aggregate([
  {"$match":
    {"amenity":{"$exists":1}}
  },
  {"$group":

```

```

        {"_id": "$amenity",
         "count": {"$sum": 1}}
    },
    {"$sort":
     {"count": -1}
    },
    {"$limit": 10}
])
/* 1 */
{
    "_id" : "recycling",
    "count" : 322
}

/* 2 */
{
    "_id" : "parking",
    "count" : 277
}

/* 3 */
{
    "_id" : "restaurant",
    "count" : 181
}

/* 4 */
{
    "_id" : "place_of_worship",
    "count" : 168
}

/* 5 */
{
    "_id" : "drinking_water",
    "count" : 136
}

/* 6 */
{
    "_id" : "bench",
    "count" : 132
}

/* 7 */
{
    "_id" : "school",
    "count" : 110
}

/* 8 */
{
    "_id" : "parking_space",
    "count" : 106
}

```

```

}

/* 9 */
{
  "_id" : "bar",
  "count" : 94
}

/* 10 */
{
  "_id" : "waste_disposal",
  "count" : 91
}

```

## 1.4 4. Other ideas about the datasets

Some ideas come to my mind when i see this dataset. Firt of all is due to the lack of information and few maintenance something must be done. For example this population was too many years ago:

```

{
  "_id" : ObjectId("56d1798800a9a619ba0e29ef"),
  "source:name" : "Nomenclátor Geográfico de Municipios y Entidades de Población",
  "source:file" : "http://centrodedescargas.cnig.es/CentroDescargas/equipamiento/BD_Municipios-Entida",
  "source:ele" : "MDT5",
  "name" : "Pámanes",
  "created" : {
    "changeset" : "9515624",
    "user" : "egrn",
    "version" : "2",
    "uid" : "399394",
    "timestamp" : "2011-10-09T18:25:42Z"
  },
  "ref:ine" : "39037000200",
  "pos" : [
    43.3566433,
    -3.773787
  ],
  "ele" : "84",
  "source:date" : "2011-06",
  "source" : "Instituto Geográfico Nacional",
  "place" : "suburb",
  "population:date" : "2009",
  "is_in:province_code" : "39",
  "type" : "node",
  "id" : "259168943",
  "population" : "749"
}

```

This kind of things made the data less powerful. So why not rewarding people that set and update the data. Private amenities like restaurants or even the city halls could give some money to maintain it.

Another thing that could be done with the dataset is trying to know some things about the cities. For example:

- Is a poor or a rich city?
- Age average

- Tourism level

All this questions maybe could be answered using machine learning techniques that i expect to learn during this udacity nanodegree program.

## 1.5 5. Appendix

All the code used to process the openstreetmap data is stored in my github account [https://github.com/jmonterrubio/Wrangle\\_Openstreetmap\\_Data](https://github.com/jmonterrubio/Wrangle_Openstreetmap_Data).

MongoDB version used to explore the data is v3.2.1 used with a management tool called [Robomongo](#).