# Data Wrangle OpenStreetMaps Data

github: https://github.com/tollek/udacity-data-science/tree/master/p3

Map Area: Cracow (Kraków / Krakow), Poland

https://www.openstreetmap.org/relation/2768922

https://mapzen.com/data/metro-extracts, city name 'Krakow'

## Problems encountered in your map

The initial pass through the Cracow OSM file has shown that this area data is very consistent. Most of the fields that would be an easy target for automatic error/inconsistency detection has been already fixed in the map. I managed to find validation & consistency issues, but there were minor, thus I decided to focus more on map completeness. I am discussing the problems in more details below.

### Validity: invalid postal codes format

Polish postal codes have format XX-YYY, were both X and Y are digits. It is quite rare to spot a postal code that does not have the '-' (hyphen). Although such postal code would be interpreted correctly, it's considered as an error.

audit_postal_codes.py script prints report about the postal codes issues:

```
# of valid postal codes:  846
# of invalid postal codes:  1

Invalid postal codes:
{'30129': '30-129'}
```

The script found only one invalid postal code, with 846 valid ones. Moreover, the problematic value occurred only once in the whole OSM file. The valid value of the problematic code (30-129) has 55 occurrences. The problem found is clearly a single entity of human error.

### Consistency: multiple entries for the same street names

Names of polish streets have different construction from US ones. Most of the street names either: - starts with 'ulica' (street) - starts with 'aleja' (alley) - don't have any road type word in their names

This fact simplifies the way street names are entered into OSM file: - if street name is 'ulica XYZ', the 'ulica' word is skipped, leaving the 'XYZ' as the street name - if the street name start with 'aleja', the whole word (without abbrevations) is put into OSM file - if street does not match any of above, whole street name is put into OSM file.

The simple rule above significantly reduces number of inconsistency errors due to different ways street names is entered into the OSM file. On the other hand, there is still some room for different types of error: - street name of format 'X Y Z' is put as 'X Y Z' and 'Y X Z' (X, Y - first and second name, Z - lastname of some historical figure) - name of person put with or without firstname - name of person with title (like 'General') and without the title

audit_street_names.py script prints report about street name inconsistencies:

```
Kamieńskiego
   Henryka Kamieńskiego, Kraków (9)
   Generała Henryka Kamieńskiego, Kraków (77)

Pokoju
   Pokoju, Kraków (4)
   Aleja Pokoju, Kraków (229)

Roweckiego
   Stefana Roweckiego, Kraków (3)
   Grota-Roweckiego, Kraków (1)
   Generała Stefana Grota-Roweckiego, Kraków (76)

Radzikowskiego
   Walerego Eljasza Radzikowskiego, Kraków (141)
   Eljasza Walerego Radzikowskiego, Kraków (28)

Słowackiego
   Juliusza Słowackiego, Kraków (5)
   Aleja Juliusza Słowackiego, Kraków (94)

Total number of inconsistent names:  52
Total number of entries with inconsistent names:  2140
Exeption list size: 352
```

The report above shows that even with clear rules about the format for street names, there are still some errors.

What is very promising about the report above, is that it gives very concrete, actionable hints about what can be done to unify map entries and remove inconsistency.

I find the result of this particular audit very interesting, although, I must underline, that there was significant time invested in marking false postitives. Total list of exceptions has 352 elements ( audit_street_names_exceptions.py). Those are street names that would be printed on report above but should not be. Algorithm that searches for inconsistency might be more sophisticated (e.g. look for X-Y-Z && Y-X-Z patterns), but that would increase computational complexity. Any further work tha tries to reuse this algorithm, should start with reviewing the list of exceptions and rewriting some of them as more specific rules (X-Y-Z && Y-X-Z, X-Y-Z && Y-Z etc.). Longer list of more specialized scenarios would probably generate most of the inconsistency issues with much less manual work involved.

### Completeness: missing restaurants

One of the most interesting data about map, is how complete data does it have. To audit OSM map completeness, I found list of recommended restaurants: http://krakow.pl/odwiedz_krakow/1410,0,0,artykul,restauracje.html. The list has been published on Cracow official internet platform. It contains multiple, well-known restaurants, that are recognized by people who live in the city for some time. What is important, most of the recommended restaurants are long-running businesses, which eliminats argument, that they're too new to be embedded in OSM.

I've created a script restaurants/extract_restaurants.py, which creates .txt file with cleaned-up list of the restaurants names.

The list of restaurants names is then use by audit_restaurant_completeness.py. Generated report:

```
Found:  40
Not found:  111
Not found:  73.5%
```

Clearly, proportion of not found restaurants is significanlty higher than found ones. Even if we take into account finding algorithm, which was not super accurate (although, acceptable after manual fixings of some of the restaurant names), the numbers are very unsatisfactory. 73.5% of missing restaurants makes the OSM an unreliable data source (at least, on its own), when it comes to analysing restaurant data. Any analysis that will use only the OSM data, should start with evaluating map completeness for given type of amenity.

## Data overview

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

```
size of the file:
519M   krakow_poland.osm
593M   krakow_poland.osm.json
```

```
number of unique users:

use osm
db.cracow.aggregate([
    {
        $group: {
            "_id": "$created.user"
        }
    },
    {
        $group: {
            "_id": "unique users",
            "count": {"$sum": 1},
        }
    },
])

Response:
{ "_id" : "unique users", "count" : 1068 }
```

```
number of nodes and ways

use osm
db.cracow.aggregate([
    {
        "$match": {
            "$or": [
                {"type": "node" },
                {"type": "way"  } ,
            ],
        },
    },
    {
        $group: {
            "_id": "$type",
            "count": {"$sum": 1},
        }
    },
])

Response:
{ "_id" : "way", "count" : 281776 }
{ "_id" : "node", "count" : 2379100 }
```

```
number of chosen type of nodes (swimming pools):

use osm
db.cracow.aggregate([
    {
        "$match": {
            "amenity": "swimming_pool" ,
        },
    },
    {
        $group: {
            "_id": "$amenity",
            "count": {"$sum": 1},
        },
    },
])

Response:
{ "_id" : "swimming_pool", "count" : 17 }
```

## Other ideas about the datasets

### Using the dataset

```
Find best streets to have fun:

use osm
db.cracow.aggregate([
    {
        $match: {
            $or: [
                {"amenity": "restaurant"},
                {"amenity": "bar"} ,
                {"amenity": "pub"} ,
                {"amenity": "cafe"} ,
            ],
            "address.street": {$exists: true},
        },
    },
    {
        $group: {
            "_id": "$address.street",
            "count": {"$sum": 1},
        },
    },
    {
        $sort: {
            "count": -1
        }
    },
    {
        $limit: 5,
    }
])


Response:
{ "_id" : "Rynek Główny", "count" : 41 }
{ "_id" : "Józefa", "count" : 15 }
{ "_id" : "Świętego Tomasza", "count" : 6 }
{ "_id" : "Szewska", "count" : 6 }
{ "_id" : "Mogilska", "count" : 5 }
```

```
Find the bank with largest number of offices:

use osm
db.cracow.aggregate([
    {
        "$match": {
            "amenity": "bank" ,
            "name": {$exists: true},
        },
    },
    {
        $group: {
            "_id": "$name",
            "count": {"$sum": 1},
        },
    },
    {
        $sort: {
            "count": -1
        }
    },
    {
        "$limit": 5,
    }
])

Response:
{ "_id" : "Millennium Bank", "count" : 17 }
{ "_id" : "Deutsche Bank", "count" : 7 }
{ "_id" : "Bank Spółdzielczy", "count" : 6 }
{ "_id" : "Alior Bank", "count" : 6 }
{ "_id" : "PKO SA", "count" : 5 }
```

## Improving the dataset

Auditing the completeness of data has shown important problem with missing restaurant data. We can look at this problems from 2 angles: - does the problem happen only for restaurants? or for private businesses in general? or is also true for state-run amenities: post offices, state agencies, etc? - how can we encourage restaurants owners to put their data into OSM?

### Completeness benchmark

Being up-to-date with all privately run businesses is super difficult task, especially for projects that are run mostly thanks to voluntary work. I checked completeness of restaurant data, but the low completeness percentage might not be a problem for different types of amenities. First of all, I expect large differences in coverage for facilities that have one central index, which can be used to pull data from. That's true e.g. for post offices or police stations. There are single pages where OSM editors can pull the data from.

Situation is completely different for private business and restuarants are only one example of such. Bars, cafes, barbers, etc. do not have one single index (actually, there migt be one, I discuss it in next section), which makes keeping the data in sync very difficult task.

Some type of live benchmark that pulls data from well defined 'sources of truth' and verifies how many of the entries can be found in OSM could be an incentive to work on amenities with low coverage. Next to the coverage and 'delta' (with red/green colors as seen in stock market charts) one could see user names which contributed most to increase of the benchmarks score.

#### Encourage business owners to publish in OSM

Adding new business to Open Street Map is quite easy task and takes just a few minutes for people who have basic computer skills. I belive that main problem is lack of awareness among business owners. Do they know that OSM exist? Do they know that they can add the business data by themselves? Do they know it's free?

In Poland every new business needs to register itself for tax & statistical purposes. Data about the businesses can be pulled out from a central database. After being pulled and cross-checked with existing OSM entries, we could build a computer bot which contacts the business owners and informs about OSM and how it's easy to add entry about a new business. There are a few challenges which would have to be tackled: - license problems: I don't know the licesne details of OSM data (if it can use such a central database) and the database itself. It uses captcha to prevent itself from bots, but maybe there is a bulk access possible for projects like OSM? - data in the central database is built for different purposes than improving online maps. There might be some cricual data missing: e.g. business email address - the communication with business owners needs to be nonintrusive.

## Conclusions

I am hugely impressed with the amount of data that have been put into the Cracow area OSM. Even with the problems I have found during the map audit, I am convinced that the map can be used as the data source for user applications. With some careful pre-analysis, it can most likely be used for analytical purposes also.

### References:

- Kraków on Open Street Map
- Kraków (Cracow) metro extract [city name 'Krakow']
- List of Cracow restaurants
- MongoDB reference