Wrangling OSM Data

Using MongoDB

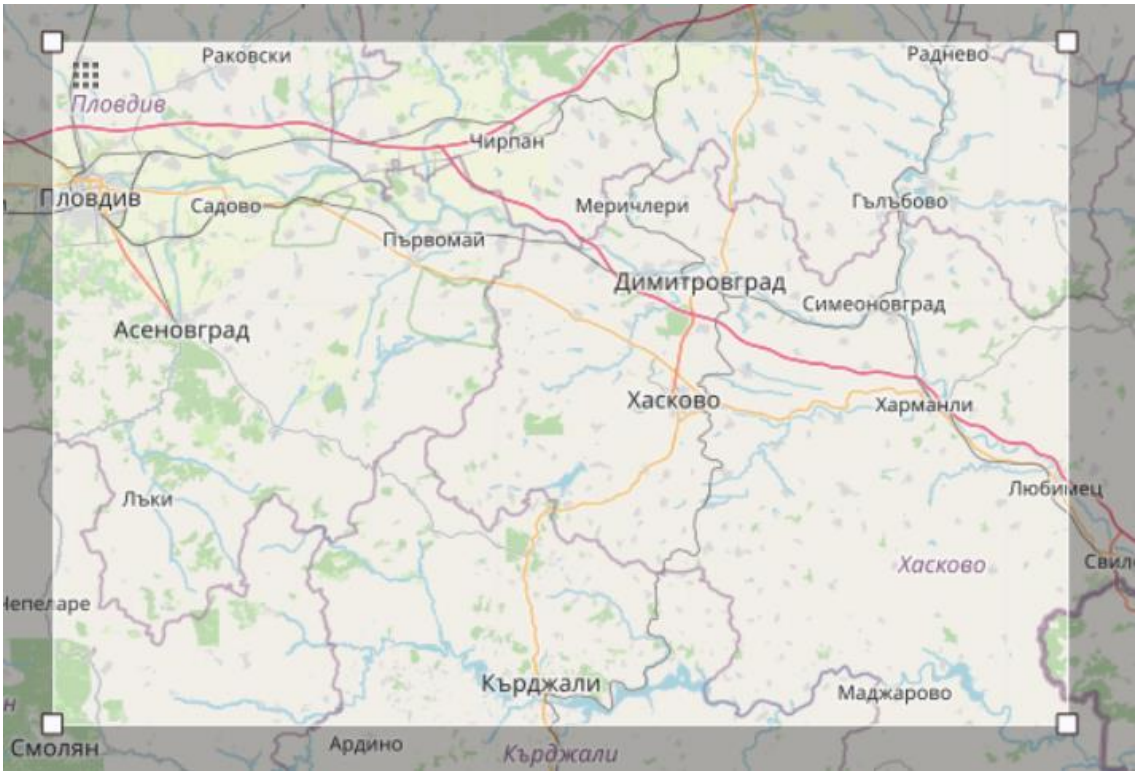
FOREWORD

This document is compiled in the context of the Airbus Data Analytics Nanodegree for the purpose of demonstrating learning for the Data Wrangling module of the course. The document and associated data are created or collected for training purpose and are only to be used for evaluation of learning in the context of this training.

1. MAP AREA

The map is of the region Martisa and Arda rivers in Bulgaria, South-Eastern Europe. It specifically include the region bound by 41.6 South, 24.7 East, 42.3 North, 26.0 West. This region contains several small cities, multiple towns and villages, as well as both industrial and agricultural objects and I chose it with the expectation of a wide variety of objects.

https://www.openstreetmap.org/export#map=9/41.9843/25.3729



Some relevant statistics for the dataset are presented below:

Statistic	Value
Size of file	227 MB (238 617 944 bytes)
Number of unique users	815
Number of nodes	1 097 260
Number of ways	120 256
Number of places without “in_in” field	318
Number of power-related entries	11 838
Number of power plants	5
Number of power generators	27
Number of restaurant	379

2. PROBLEMS IN THE DATASET

This section outlines problems or potential problems were identified in the dataset. Any necessary actions have been also detailed. The auditing of the dataset has been performed in parallel with parsing the dataset from XML (as downloaded from OSM) to JSON (to be ready for import in MongoDB).

2.1. Use of Cyrillic script

Since the data is from Bulgaria, where the local script is different then the Roman alphabet, a large amount of the data contained in the export uses Cyrillic script. Since Python 3 provides very good support for UTF-8 formatting, little action was needed during cleaning and processing the data. However, UTF-8 support should be a consideration for any downstream tools using this data.

```
<way changeset="11570299" id="28646584" timestamp="2012-05-11T19:03:27Z"
uid="145231" user="woodpeck_repair" version="5">
    <tag k="addr:city" v="Кърджали" />
    <tag k="addr:country" v="BG" />
    <tag k="addr:postcode" v="6600" />
    <tag k="addr:street" v="Тина Киркова" />
    <tag k="building" v="yes" />
    <tag k="building:levels" v="1" />
    <tag k="is_in" v="Bulgaria" />
    <tag k="source" v="bgtopomaps" />
</way>
```

To properly write JSON to a utf-8 encoded file with Python 3, the correct encoding needs to be ensured which can be accomplished using the codecs module. Similar care needs to be taken also when reading a file. (https://stackoverflow.com/questions/18337407/saving-utf-8-texts-in-json-dumps-as-utf8-not-as-u-escape-sequence):

```
with codecs.open(filename, "w", encoding="utf-8") as of:
        of.write(json.dumps(data, indent=2, ensure_ascii=False)+"\n")
```

2.2. Name variants

Multiple variants and spellings of the name of each location were sometimes available. This is as per OSM convention to provide fully international support for multiple special cases which are possible in naming entities https://wiki.openstreetmap.org/wiki/Key:name. For the purpose of this analysis, this has been simplified to include only the local name and an English translation if available.

XML:

```
<node changeset="73229130" id="273488963" lat="42.0582493" lon="25.5916466"
timestamp="2019-08-10T20:53:16Z" uid="150201" user="plamen" version="17">
    <tag k="ekatte" v="21052" />
    <tag k="int_name" v="Dimitrovgrad" />
    <tag k="is_in:country" v="България" />
    <tag k="is_in:region" v="Хасково" />
    <tag k="name" v="Димитровград" />
    <tag k="name:bg" v="Димитровград" />
    <tag k="name:de" v="Dimitrowgrad" />
    <tag k="name:el" v="Ντιμίτροβγκραντ" />
    <tag k="name:en" v="Dimitrovgrad" />
    <tag k="name:pl" v="Dimitrowgrad" />
    <tag k="name:ro" v="Dimitrovgrad" />
    <tag k="name:tr" v="Kayacık" />
    <tag k="place" v="city" />
    <tag k="population" v="49061" />
```

```
    <tag k="source" v="bgtopomaps" />
    <tag k="wikidata" v="Q331048" />
    <tag k="wikipedia" v="en:Dimitrovgrad, Bulgaria" />
</node>
```

JSON:

```
{
    "class": "node",
    "pos": [
      42.0582493,
      25.5916466
    ],
    "created": {
      "changeset": "73229130",
      "timestamp": "2019-08-10T20:53:16Z",
      "uid": "150201",
      "user": "plamen",
      "version": "17"
    },
    "id": "273488963",
    "ekatte": "21052",
    "name": {
      "English": "Dimitrovgrad",
      "local": "Димитровград"
    },
    "is_in": {
      "country": "България",
      "region": "Хасково"
    },
    "place": "city",
    "population": "49061",
    "source": "bgtopomaps",
    "wikidata": "Q331048"
}
```

2.3. OSM Namespace notation

OSM uses namespaces https://wiki.openstreetmap.org/wiki/Namespace for further subdivision of possible keys in a tag. The namespace convention however is not directly compatible with the No-SQL schema used in MongoDB, specifically it is problematic when the value occurs both with and without a namespace suffix. A document in MongoDB can refer to either a string (e.g. “warehouse”) or another document, but not to both. To handle the generic case of a nested namespace, all keys where a namespace is detected are converted to a document. The schema adds different levels of documents for the various suffixed namespaces, as shown in the example XML and JSON documents below (specifically, the “building” field has been moved to the “type” subfield in the “building” document).

XML:

```
<way changeset="28934488" id="103474461" timestamp="2015-02-18T14:36:59Z"
uid="172013" user="balkanyug" version="3">
    <tag k="addr:housenumber" v="36" />
    <tag k="addr:street" v="бул.Съединение" />
    <tag k="building" v="warehouse" />
    <tag k="building:levels" v="2" />
    <tag k="name" v="РКС Хасково" />
</way>
```

JSON:

```
{
    "class": "way",
    "created": {
      "changeset": "28934488",
      "timestamp": "2015-02-18T14:36:59Z",
      "uid": "172013",
      "user": "balkanyug",
      "version": "3"
    },
    "id": "103474461",
    "address": {
      "housenumber": "36",
      "street": "бул. Съединение"
    },
    "building": {
      "type": "warehouse",
      "levels": "2"
    },
    "name": {
      "local": "РКС Хасково"
    }
}
```


2.4. Address Notation

As one of the specific subcases of namespace notation, the “addr” field and associated suffixes have been converted to the “address” document during JSON conversion, as also illustrated on the example above.

2.5. Street Naming

Street names have been audited to the following requirements. When a name is not conforming, the value has been updated using *audit_address.py*:

- All street names should be in Cyrillic
- All street names should follow proper cases with the exception of the prefix.
- The street prefix should follow abbreviated form and start with a lower case letter as per http://www.upu.int/fileadmin/documentsFiles/activities/addressingUnit/bgrEn.pdf
- Several typos have also been identified and corrected

3. DATA EXPLORATION

Several study questions are answered in the following sections.

3.1. What is the most frequently used type of power generation?

The OSM data contains several fields relevant to power. To answer the question above, two separate fields are relevant – power plants and power generators. The type of power is stored into a subfield “source”. Since two different types of fields are used, “$or” and “$isNull” operators have been utilized to ensure the result is obtained from a single query.

```
pipeline = [{"$match": {"$or":[ {"power": 'generator'},
                                 {"power": 'plant'}]}},
            {"$project":  {"power": 1,
                           "source": {"$ifNull": ["$generator.source",
                                                  "$plant.source"]}}},
            {"$group": { "_id": "$source",
                         "count": {"$sum": 1}}},
            {"$sort": {"count": -1}}]
```

The result are shown in the table below:

Power source	Number
Solar	17
Hydro	5
Coal	5
Gas	1
Unknown	4

Some plants and generators do not have a listed source, but the most common type of power source is solar, followed by hydro and coal with equal counts. While this may suggest the energy generation is very green, a further query reveals that coal plants tend to have much higher energy outputs per generator compared to hydro or solar. However, since not enough entries have the needed data (stored in the “generator.output.electricity” field), an accumulation was not carried out.

3.2. What is the type of entity which most often has a listed website?

Internet is growing in importance as a key way for people to find information about their surroundings. For many entities, websites are an important way to communicate with potential customers and storing websites in OSM is a convenient connection between the location and relevant information. The query below combines several types of objects (shops, amenities, tourism-related, offices, etc.) and groups them if they have a listed website.

```
pipeline = [{"$match": {"class": "node",
                        "website": EXISTS}},
            {"$project": {"entity": {"$switch": {"branches": [
                    {"case": NOT_NULL("$amenity"),
                     "then": "$amenity"},
                    {"case": NOT_NULL("$shop"),
                     "then": {"$concat": [{"$cond": [{"$eq": ["$shop",
                                                              'yes']},
                                                     "",
                                                     "$shop"]},
                                                " ",
                                                "shop"]}},
                    {"case": NOT_NULL("$place"),
                     "then": "$place"},
                    {"case": NOT_NULL("$tourism"),
                     "then": "$tourism"},
                    {"case": NOT_NULL("$office"),
                     "then": "office"}],
                "default": None}}}},
            {"$group": { "_id": "$entity",
                         "count": {"$sum": 1}}},
            {"$match": {"count": {"$gte": 5}}},
            {"$sort": {"count": -1}}]
```

The type of entity with most listed websites is restaurant (24), followed by clothes shops (21), and offices (19). However further exploration reveals that 352 restaurants do not have a listed website. It is also identified that multiple objects (73) have a website, but are not properly classified;

4. IDEAS FOR IMPROVEMENT

Here possible ideas for improvements of the dataset are presented. Some of them can be extended not only to this particular dataset, but to a large portion of the OSM data, and can potentially be implemented as cleaning bots.

4.1. Use positional information to augment “is_in” field

The “is_in” field outlines the relationship between various entities belonging to another entity, e.g. a city being part of a municipality, which is turn part of region, country, etc. A possible improvement is an auditing routine which completes the field with all relevant entities based on the positional information (latitude and longitude) of each node and the boundaries of each of the higher-level entities (usually present as a “relation”). This will allow for easier determination of various factors which rely on the hierarchical structure of geographical entities.

4.2. Augment addresses data using street information

In the current dataset, the address entities are often associated with a specific “node”. Many of these “nodes” often belong to a certain “way”, which can represent e.g. a street. It is common sense that the “node” addresses should share a common street, however this is not enforced in the dataset. Some available “nodes” even do not have such street information associated, which can easily be added using the “nd_ref” of the “way” and the “id” of the node to find it’s corresponding street.

4.3. Classify objects with a website

As discussed in section 3.2, multiple objects were identified where the website has been stored, but the object is poorly classified. It is possible to scrape the listed website for various keywords which would help to classify the data. A potential approach would be to use properly classified entities with websites to train a machine learning algorithm. One important consideration to such an approach is the available amount of relevant data. In this specific case, various information is likely stored in Bulgarian language, and due to the limited size of Bulgaria (the dataset used here is estimate to include 10% to 15% of the entire available data for the country), this might not be enough for machine training.