Data Analyst Nanodegree Project 3 - Data Wrangle OpenStreetMaps Data

Allan MacGowan
July 29, 2015

1. Overview of the Dataset

OpenStreetMaps Dataset:
Mississauga, Ontario, Canada (part of Toronto)
https://s3.amazonaws.com/metro-extracts.mapzen.com/mississauga_canada.osm.bz2

File sizes:
06/25/2015 12:48 AM 145,071,175 mississauga_canada.osm
07/22/2015 06:05 PM 161,117,737 mississauga_canada.osm.json

File was imported into MongoDB under a database called 'project' and collection called 'maps'.

```
# Number of documents
    > print db.maps.find().count()
    680451

# Number of nodes
    > print db.maps.find({"type":"node"}).count()
    578022

# Number of ways
    > print db.maps.find({"type":"way"}).count()
    101671

# Number of unique users
    > print len(db.maps.distinct("created.user"))
    365
```

```
# Top three contributing users
      > db.maps.aggregate([{"$group":{"_id":"$created.user",
                                      "count":{"$sum":1}}},
                           {"$sort":{"count":-1}},
                           {"$limit": 3}])
      {u'_id': u'andrewpmk', u'count': 308212}
      {u'_id': u'Victor Bielawski', u'count': 160129}
      {u'_id': u'Kevo', u'count': 74120}

# Five users have together contributed over 90% of the entries.
      User andrewpmk has contributed 45.3 %
      User Victor Bielawski has contributed 23.53 %
      User Kevo has contributed 10.89 %
      User Bootprint has contributed 6.31 %
      User Gerit Wagner has contributed 5.0 %

# Number of users with only one entry
      > db.maps.aggregate([{"$group":{"_id":"$created.user",
                                      "count":{"$sum":1}}},
                          {"$group": {"_id":"$count",
                                      "num_users":{"$sum":1}}},
                          {"$sort":{"_id":1}},
                          {"$limit": 1}])
      {u'_id': 1, u'num_users': 82}

# Format for 'k' values of the tags to ensure they can be valid keys in
MongoDB:
{'lower': 347338, 'lower_colon': 281261, 'other': 17077, 'problemchars': 0}
```

2. Problems Encountered with the Map

Below is a summary of problems with the map. The first two bullets refer to isolated issues affecting very few documents, relatively speaking. The latter two issues are more systemic.

- Some document node street names are abbreviated, incomplete or incorrect.

- One of the documents has a field for ‘contact:website’ that has a phone number as its value.
- Postal code format is inconsistent across documents. The dominant and conventional format is ‘A1A 1A1’ but some documents use ‘A1A1A1’ instead.
- Addresses for many documents contain a ‘state’ field, whereas Canada has no states, only provinces.

Street Names

The following street names were identified as being either abbreviated, incomplete or incorrect. Corrections were made before the JSON file was created.

```
mapping = { "abbreviated": {"Rd.": "Road", "St.": "Street"},
          "incorrect": {"Foster": "Foster Crescent",
                        "Winston Churchill": "Winston Churchill Boulevard",
                        "The Keanegate": "The Keanegate Street",
                        "Meyerside Drive, Unit 9": "Meyerside Drive",
                        "restaurant": "Mississauga Road"}}
```

An additional correction was needed from MongoDB:

```
   {u'_id': ObjectId('559c87a29bb2c25b2b5e8900'),
    u'address': {u'street': u'1134'},
    u'created': {u'changeset': u'26445532',
                 u'timestamp': u'2014-10-30T18:53:55Z',
                 … }

# Remove entire address field
       > db.maps.update({"address.street":"1134"}, {"$unset":{"address":"
"}})
```

Incorrect Website

A check of all documents with a website was performed and an incorrect entry was identified and corrected.

```
# View all documents with a website listing to see the incorrect entry.
        > db.maps.find({"contact:website":{"$exists": 1}})
        …
        u'http://www.spoonandfork.ca/spoon-and-fork-queensway/welcome/'
        u'+1 905-891-9300'
        u'www.bellesbeaussalonspa.com'
        …
# Correct this value by removing this field.
        > db.maps.update({"contact:website":"+1 905-891-9300"},
                {"$unset":{"contact:website":""}})
```

Postal Code Format

The postal code format is inconsistent as the following queries show.

```
# Total number of entries with postal codes.
        > db.maps.find({"address.postcode":{"$exists":1}}).count()
        775

# Total number of entries with postal code format 'A1A 1A1'.
        > db.maps.find({"address.postcode":
                        {"$regex":"[A-Z][0-9][A-Z]\s[0-9][A-Z][0-9]"}})
        729

# Total number of entries with postal code format 'A1A1A1'.
        > db.maps.find({"address.postcode":
                        {"$regex":"[A-Z][0-9][A-Z][0-9][A-Z][0-9]"}})
        45
```

To modify all postal codes with format 'A1A1A1' to 'A1A 1A1', I created the list bad_pc of all codes with format 'A1A1A1' and iterated through, updating each one.

```
    for item in bad_pc:
        db.maps.update({"address.postcode": item},
                {"$set":{"address.postcode":item[:3] + " " + item[3:]}})
```

Running the queries again:

Number of documents with postal codes in format A1A1A1: 0
Number of documents with postal codes in format A1A 1A1: 774

State Field

While most entries correctly used the ‘province’ field in the address, some used the ‘state’ field, which is not applicable for Canada.

```
# Search for entries that use the ‘state’ field.
      > db.maps.find({"address.state":{"$exists":1}}).count()
      34076

# Search for entries that use the ‘province’ field.
      > db.maps.find({"address.province":{"$exists":1}}).count()
      1541

# Update all "address.state" fields to "address.province".

   db.maps.update({"address.state":{"$exists":1}},
            {"$set":{"address.province":"ON"}},
            multi=True)

   db.maps.update({"address.state":{"$exists":1}},
            {"$unset":{"address.state":""}},
            multi=True)
```

Number of documents with addresses containing a field for state: 0
Number of documents with addresses containing a field for province: 35616

3. Other Ideas About the Dataset

Incompleteness of the Dataset

Nodes with addresses that contain a unit field are sometimes incomplete. These nodes should also contain address fields for house number, street, city, province, and postal code, but 47 of them have 5 or fewer address fields.

There are only 53 places identified as selling coffee, all for $1.50, which is clearly both incomplete and inaccurate.

As indicated earlier, approximately 90% of the entries have been submitted by only five users, and over 20% of contributors (82 of 365) have only supplied one entry.

Ways to Improve Data Quality Within OSM

Completeness of the OSM dataset for Mississauga is at least partially contingent on attracting more contributors. I would suggest gamification techniques, such as awarding badges, posting a highly visible leaderboard or establishing an awards program. Increasing breadth and depth of user participation would help fill in missing gaps and improve accuracy of the data, but there is a risk that a prominent gamification competition could clutter up the interface and deter people from using OSM.

One of the issues I encountered was the inconsistent use of the 'address:unit' field. The string 'Unit <number>' appears in the 'address:street' field, 'note' field, and 'description' field. The 'address:unit' field is seldom populated. To resolve this, one approach is to impute the missing 'address:unit' values from note, description or street values within the same node. Of course, with this approach there is a risk of incorrectly referencing other values in order to populate a given field.

Another way to improve the quality of OSM data is to encourage the use of efficient and correctly designed bots to make automated edits on a regular basis. Again, gamification can play a role here, as well as enhancing the OSM editing API for those building scripts and bots.

Additional Facts about the Dataset

Location of HOV lanes, which are useful for commuters as well as travelers to special events, such as the Pan American Games, which were recently held in Toronto and made extensive use of HOV lanes to improve traffic flow.

```
> db.maps.find({"hov:lanes":{"$exists": 1}})
u'designated||||'
u'designated||'
u'||designated'
u'designated|||'
u'designated||||'
…

# Top ten amenities
> db.maps.aggregate([{"$match":{"amenity":{"$exists":1}}},
                     {"$group":{"_id":"$amenity",
                      "count":{"$sum":1}}},
                     {"$sort":{"count":-1}},
                     {"$limit":10}])
{u'_id': u'parking', u'count': 3595}
{u'_id': u'fast_food', u'count': 441}
{u'_id': u'restaurant', u'count': 397}
{u'_id': u'school', u'count': 333}
{u'_id': u'post_box', u'count': 191}
{u'_id': u'place_of_worship', u'count': 179}
{u'_id': u'cafe', u'count': 175}
{u'_id': u'fuel', u'count': 163}
{u'_id': u'bank', u'count': 153}
{u'_id': u'bench', u'count': 125}

# Most popular fast food restaurants
> db.maps.aggregate([{"$match":{"amenity":{"$exists":1},
                       "amenity":"fast_food",
                       "cuisine":{"$exists":1}}},
                     {"$group":{"_id":"$cuisine",
                              "count":{"$sum":1}}},
                     {"$sort":{"count":-1}},
                     {"$limit":3}])
{u'_id': u'burger', u'count': 72}
{u'_id': u'sandwich', u'count': 56}
{u'_id': u'pizza', u'count': 50}
```

Query for most frequently occurring banks, which revealed inconsistencies and errors in bank names (e.g. ‘Scoitabank’ vs ‘Scotiabank’) that need to be corrected.

```
> db.maps.aggregate([{"$match":{"amenity":{"$exists":1},
                                "amenity":"bank"}},
                   {"$group":{"_id":"$name",
                                "count":{"$sum":1}}},
                   {"$sort":{"count":-1}}])
{u'_id': u'TD Canada Trust', u'count': 40}
{u'_id': u'Scotiabank', u'count': 18}
{u'_id': u'RBC', u'count': 17}
{u'_id': u'CIBC', u'count': 17}
{u'_id': u'BMO Bank of Montreal', u'count': 10}
{u'_id': u'Bank of Montreal', u'count': 6}
{u'_id': u'CIBC Banking Centre', u'count': 6}
{u'_id': u'RBC Financial Group', u'count': 5}
{u'_id': u'BMO', u'count': 5}
…
{u'_id': u'Scoitabank', u'count': 1}
```

Top five most occurring Christian denominations, which also revealed an inconsistency in the denomination field (Catholic and Roman Catholic)

```
> db.maps.aggregate([{"$match":{"amenity":{"$exists":1},
                                "amenity":"place_of_worship",
                                "religion":"christian",
                                "denomination":{"$exists":1}}},
                   {"$group":{"_id":"$denomination",
                                "count":{"$sum":1}}},
                   {"$sort":{"count":-1}},
                     {"$limit":5}])
{u'_id': u'catholic', u'count': 19}
{u'_id': u'united', u'count': 19}
{u'_id': u'baptist', u'count': 16}
{u'_id': u'roman_catholic', u'count': 15}
{u'_id': u'presbyterian', u'count': 14}
```

4. Conclusion

This analysis shows that this dataset is incomplete and fairly ‘dirty’, with many inconsistencies and errors across multiple fields. Given that most of the data has been provided by only a few users, it would be beneficial to run a campaign to recruit new contributors and increase the number of entries per user. It may also be useful to create templates with designated options for specific fields to alleviate inconsistencies.