Wrangle OpenStreetMap Data

Map Area

Boston, MA, USA

- https://www.openstreetmap.org/relation/2315704
- https://mapzen.com/data/metro-extracts/metro/boston_massachusetts/

Downloaded on December 1, 2016

This map is from my city of birth: I was born there while my parents (french father, swedish mother) were foreign students at Harvard University before returning to Europe.

"boston_massachusetts_sample.osm" is a shortened version (53 MB) of the original 424 MB file.

Problems encountered in the Map and cleaning steps

a. Incorrect street abbreviations

Such as 'St' in 'Main St' instead of 'Street'.

Cleaning step:

I used the USPD Street Suffix Abbreviations list as reference for mapping.
Then I audited the street types to detect errors and applied corrections based on mapping.

```
# Function to correct street names using wrong suffix
def update_name(name, mapping):
    """ Substitutes incorrect abbreviation with correct one. """
    m = street_type_re.search(name)
    if m:
        street_type = m.group()

        temp= 0
        try:
            temp = int(street_type)
        except:
            pass

        if street_type not in expected and temp == 0:
            try:
                name = re.sub(street_type_re, mapping[street_type], name)
            except:
                pass
    return name
```

b. Incorrect postal codes.

US postal codes follow the 5-digit format like '02118' which may lead to different errors.

- Some codes differed due to additional signs such as 'MA 02118' or '02136-2460'.

- Also some postal codes were not be compatible with the Boston, MA area codes who start with "01xxx" or "02xxx".

Such as '03079' (Salem in New Hampshire) or they might represent some other entry, such as '(617) 495-1000' is a phone number (Harvard U.).

Cleaning step:

When codes were incorrectly formated with additional signs, I removed those to fit the 5-digit format.

When codes did not belong to Boston MA area, they were removed.

```
# Function to correct format of postal codes
def update_zip(post_code):
    """ Extracts 5-digit postal codes from postal codes in different formats
        and deletes postal codes that do not correspond to Massachussets area.
    """

    if post_code[0:2] == 'Ma' or post_code[0:2] == 'MA':
        post_code = post_code[3:].strip()

    if len(post_code) >5 and len(post_code) == 10 and (post_code[0:2] == '01' or
post_code[0:2] == '02'):
        post_code = post_code[0:5]
    elif len(post_code) < 5 or (post_code[0:2] != '01' and post_code[0:2] !=
'02') :
        post_code = ''
    elif len(post_code) > 5 and post_code[5]==' ':
        post_code = post_code[0:5]
    elif len(post_code)>5:
        post_code = ''

    return post_code
```

c. Incorrect abbreviation of state entries.

All state entries should either contain the official abbreviation for Massachusetts, 'MA', as opposed to 'Mass', 'mass', 'ma', and 'Ma' to 'MA';

or an empty string, if the state entry is not related to 'MA', such as 'New York', 'NH', ...

Cleaning step:

When entries are a mispell variation of MA such as 'ma', 'Ma', 'Mass' etc., they were corrected with 'MA'.

Otherwise, like "NY", they were removed.

```
# Function to correct state entries
def update_state(state):
    """ Deletes U.S. state entries not related to Massachussets and formats all
remaining to 'MA'."""

    if state.startswith('ma') or state.startswith('Ma') or
state.startswith('MA') or state == 'M':
        state = 'MA'
    else:
        state = ''

    return state
```

Data Overview

The following SQL queries give a general summary of the data as well as some other interesting facts.

File sizes

boston_massachusetts.osm ..... 424 MB
boston_massachusetts_sample.osm .... 53 MB
BostonMA.db ..... 31 MB
nodes.csv ..... 19 MB
nodes_tags.csv ..... 7 MB
ways.csv ..... 3 MB
ways_nodes.csv ..... 7 MB
ways_tags.csv ..... 3 MB

Number of Nodes

```
query = "SELECT count(*) FROM nodes;"
```

241,256

Number of Ways

```
query = "SELECT count(*) FROM ways;"
```

38,598

Number of Unique Users

```
query = "SELECT count(DISTINCT(temp.uid)) FROM (SELECT user, uid FROM ways UNION
ALL SELECT user, uid FROM nodes) as temp;"
```

679

Top 10 contributors

```
query = "SELECT temp.user, count(*) as posts
FROM (SELECT user, uid FROM ways UNION ALL SELECT user, uid FROM nodes) as temp
GROUP BY temp.user ORDER BY posts DESC LIMIT 10;"
```

crschmidt, 150535
jremillard-massgis, 53877
OceanVortex, 11457
wambag, 10019
morganwahl, 8757
ryebread, 8405)
MassGIS Import, 7920
ingalls_imports, 4058
Ahlzen, 3364
mapper999, 1891

The contributions of the Top 10 users is incredibly skewed as they weight over 92% of total contributions [Top 10 / (nodes + ways)].

Top 20 Amenities

```
query = "SELECT temp.value, count(*) as num
FROM (SELECT key,value FROM ways_tags UNION ALL SELECT key,value FROM
nodes_tags) as temp
WHERE temp.key='amenity' GROUP BY temp.value ORDER BY num DESC LIMIT 20;"
```

parking, 166
bench, 138
school, 96
restaurant, 79
parking_space, 57
place_of_worship, 56
library, 38
cafe, 33
bicycle_parking, 32
fast_food, 26
bicycle_rental, 17
pub, 13
university, 13
fire_station, 12
bar, 11
hospital, 11
bank, 10
post_box, 10
fountain, 9
fuel, 9

Additional Ideas

As I currently live in Stockholm, Sweden, I took note of the massive integration of wheelchair accessibility into public structures and amenities.

So I searched for information available on wheelchair accessibility and compared it with the number of amenities.

Number of Wheelchair Accessibility information

```
query = "SELECT count(*) FROM (SELECT key,value FROM ways_tags UNION ALL SELECT key,value FROM nodes_tags) WHERE key='wheelchair';"
```

32

Number of Amenities

```
query = "SELECT count(*) FROM (SELECT key,value FROM ways_tags UNION ALL SELECT key,value FROM nodes_tags) WHERE key='amenity';"
```

950

This ratio 32 / 950 = 3.37% is extremely poor and should be explored in more depths.

A follow-up project could focus on a verification of:

(1) what part of the bad performance is due to missing information, ie. how many amenities DO HAVE wheelchair accessibility but that information went MIA,

(2) and what part is reflective of the true situation, ie. many amenities DO NOT HAVE wheelchair accessibility.

Benefits: beside the obvious social benefits and duties for helping persons with restrained mobility, there are business opportunities for commerces (shops, restaurants, hotels) who act accordingly.

Problems: the follow-up project might require a lot of minute manual work to verify the missing information if not available in a compatible database.