OpenStreetMap Data Case Study

Map Area

Nashville, TN, United States

https://mapzen.com/data/metro-extracts/

https://www.openstreetmap.org/node/38451000

I'm a graduate student of Vanderbilt University. I have been there for one year. Exploring the map data of Nashville is an interesting thing for me.

Problems Encountered in the Map

First of all, let's have a glimpse of our tag in the dataset. I create two dictionaries. One for showing how many unique key we have in tag. One for showing unique value of each key. Result is shown as below:

```
[('highway', 199724),
 ('name', 131584),
 ('tiger:county', 128021),
 ('tiger:cfcc', 127954),
 ('tiger:reviewed', 122108),
 ('tiger:name_base', 89391),
 ('tiger:name_type', 84276),
 ('tiger:zip_left', 64598),
 ('tiger:zip_right', 59440),
 ('power', 42823),
 ('building', 40265),
 ('tiger:tlid', 34214),
 ('tiger:source', 34144),
 ('tiger:upload_uuid', 33877),
 ('tiger:separated', 22278),
 ('ele', 21312),
 ('access', 21269),
 ……..]
```

For specific key, we can inspect it as belown:

```
key_value['addr:city']
{'Adams',
 'Antioch',
 'Antler, Tennessee',
 'Bell Buckle',
 'Brentwood',
 'Burns',
 'Cane Ridge',
 'Centerville',
 'Christiana',
 'Clarksville',
 ……..}
```

According to the result, we have 999 unique keys. All the tags need to be cleaned, since we don't trust these data. But cleaning them one by one will be a huge work, thus I decide to import part of them into our database,

Here is so problem I find in this dataset.

1.'addr:street'

There exist repetition in the street name. For example, 'Dr' and 'Drive'. Some of these name are not capitalized: 'avenue' and 'Avenue'.

To solve this problem, we create a dictionary and a function to map wrong word into correct way. For example:

'ave' :'Avenue',

'avenue' : 'Avenue',

'Ave' : 'Avenue',

…..

```
def update_street(name, mapping):
    name_split = name.split(' ')
    if name_split[-1] in mapping:
        name_split[-1] = mapping[name_split[-1]]
        name = " ".join(name_split)

    return name
```

2.'addr:city'

Some of the city name are spelled in the form of "city, state" or "city-county". There also exist repetition: 'Thompson"s Station', "Thompson's Station". Some of them need to be capitalized. For some city name, like 'LaVergne', blank is missing. Furthermore, there is spelling mistake, 'Mount Joliet' should be 'Mount Juliet'.

```
def update_city(city):
    city = city.split(',')[0]#2
    city = city.split('-')[0]#2
    city = city[0].upper() + city[1:]
    city = city.replace("\"" ,"" )#3
    city = city.replace("'" , "")#3
    if 'Joliet' in city:
        city = city.replace('Joliet','Juliet')#4

    before = None
    index = 0
    for letter in city:
        if not before:
            before = letter
        elif letter.isupper() and before != " ":
            city = city[0:index]+ " "+city[index:]
            index +=1
        else:
            before = letter
        index +=1
    return city
```

3.'addr:state'

The value of state is shown as below:

```
{'37042', 'KY', 'TB', 'TN', 'Tennessee', 'tn'}
```

It's clearly that all of them should be set as 'TN' except 'KY'.

```
def update_state(state):
    if state != "KY":
        state = "TN"
    return state
```

4. 'addr:postcode'

We want all postcode are 5-digit number, but there exist some problem like that:

```
'37243-0471', 'TN','TN 37203'
```

The goal is set all of them to be 5-digit number.

```
def update_postcode(postcode):
    postcode = postcode.split('-')[0]
    if (len( postcode.split(' '))>1):
        postcode = postcode.split(' ')[1]
    if postcode=='TN':
        postcode = None
    return postcode
```

For each problem I described, I create a function. Then I combine these function to a function called ‘update’. When I create csv file, the ‘update’ function will be used.

```
def update(task,value,mapping):
    if task =='state':
        value = update_state(value)
    elif task == 'city':
        value = update_city(value)
    elif  task == 'street':
        value = update_street(value, mapping)
    elif task == 'postcode':
        value = update_postcode(value)
    return value
```

Explore Database

When we create a csv file, we need to import our csv into database.

.mode csv

.import nodes.csv nodes

.import nodes_tags.csv nodes_tags

.import ways.csv ways

.import ways_nodes.csv ways_nodes

.import ways_tags.csv ways_tags

1.Size of file

```
nashville_tennessee.osm ......... 258 MB
nashville.db .......... 162 MB
nodes.csv ............. 97 MB
nodes_tags.csv ........ 141 KB
ways.csv .............. 7.08 MB
ways_tags.csv ......... 1.90 MB
ways_nodes.cv ......... 31.8 MB
```

2. Unique users

```
query = '''
select count(distinct uid) as unique_num
from (
select uid from nodes
union all
select uid from ways
)
'''
```

There are 853 unique users

3. Nodes and ways

```
query = '''
select count(*) as num
from nodes
'''
```

```
query = '''
select count(*) as num
from ways
'''
```

There are 1206240 nodes and 123097 ways.

3. Tags

```
query = '''
select count(*) as num, count(distinct key) as keys
from (
select * from nodes_tags union all
select * from ways_tags)
'''
```

There are 63894 tags and 6 unique keys.

4. 10 cities name occurred most

```
query ='''
select value ,count(value) as num
from (
select * from nodes_tags union all
select * from ways_tags)
where key = 'city'
group by value
order by num desc
limit 10
'''
```

```
[(u'Clarksville', 6832),
 (u'Franklin', 542),
 (u'Murfreesboro', 250),
 (u'Brentwood', 218),
 (u'Nashville', 188),
 (u'Columbia', 19),
 (u'Nolensville', 17),
 (u'Springfield', 15),
 (u'Spring Hill', 11),
 (u'Mc Minnville', 8)]
```

Additional Ideas

Additional Data Exploration

1. This access condition in Nashville

```
query = '''
select value ,count (value) as num
from (
select * from nodes_tags union all
select * from ways_tags)
where key = 'access'
group by value
order by num desc
limit 10

'''
```

```
[(u'private', 9898),
 (u'customers', 190),
 (u'no', 146),
 (u'yes', 139),
 (u'destination', 121),
 (u'permissive', 57),
 (u'agricultural', 30),
 (u'public', 29),
 (u'delivery', 11),
 (u'designated', 5)]
```

2. 10 building occurred least in Nashville

```
query ='''
select value ,count(value) as num
from (
select * from nodes_tags union all
select * from ways_tags)
where key = 'building'
group by value
order by num
limit 10
'''
```

```
[(u'Condemned', 1),
 (u'Recording_Industry,_Gem and Fossil', 1),
 (u'Station', 1),
 (u'abandoned', 1),
 (u'amphitheatre', 1),
 (u'bunker', 1),
 (u'college', 1),
 (u'manufacture', 1),
 (u'museum', 1),
 (u'no', 1)]
```

Conclusion

The tag in the map of Nashville is much complex than I thought. In this project, I only clean part of them. The more reliable data need us to spend more time. I find there is a group of tag starting with “tiger:” which are seem like cleaner than others, but I can’t sure if it is clean actually. Meanwhile, this tag group occurred most in our dataset. Thus, to improve our result, we need to analyze this later.

Furthermore, using the clean data we got before, we can draw a traveling map. In this map, we can show the type of building in different street with its access status. According to the result, we can color different region via the density of a specific requirement. For example, the more public building where visitors may interested, the darker color will show in the map.

More specifically, we can meet our goal using machine learning. First of all, we give each building an unknown parameter: attraction level. This is a value which can reflect a building's attraction to tourist. To get the value, we can collect tourist number of different building in Nashville or other city in past years. Then we can build a machine learning model, like logistic regression, to estimate the attraction level of building. After getting the result, we can predict the attraction level of every building in Nashville and show it in different color.

The main difficulty of this problem is how to collect our training data. We can get correct prediction only when we get accurate data. Like many statistic problem, there may exist a website show the number we need. The second one is when we fit our data into machine learning model, how to choose ab appropriate model is critical and even though we choose the correct model, it always has risk of overfitting.

To realize this function, we need to get the table which can reflect the relationship between public building and different street and we also need to rank different building according to its attraction to visitors. Finally, the problem we may meet is we need to write a function which can color the map according to the data we got. How to color the map can influent our traveling guide directly.