Map Area: Bangalore, India

Map Area: Bangalore, India

http://www.openstreetmap.org/way/151676284

I chose Sarjapura,Bengaluru for my analysis as I knew that the dataset will be large enough to do analysis and it will have opportunities on data wrangling.

I am currently living here.

Is there a list of Web sites, books, forums, blog posts, github repositories etc that you referred to or used in this submission (Add N/A if you did not use such resources)? N/A

Problems encountered in your map

The dataset downloaded was for Bangalore. I ran the audit.py program and found very few issues :

1. There were a few upper case street types as well as lower case types that contributed to duplicate entries like:

Code:

- 'Cantonment': set(['Bangalore Cantonment']),
- 'Circle': set(['Aurobindo Circle',
- 'H Siddiah Circle',
- 'Seeta Circle',
- 'Shoolay Circle',
- 'Siddalingaiah Circle']),
- 'College': set(['Ambedkar College']),
- 'Complex': set(['Nagarabhavi BDA Complex']),
- 'Cross': set(['Kenchapura Cross', 'Mallathalli Cross']),
- 'Gate': set(['BMTC Depot-12 Gate', 'Lakshmisagar Gate']),
- 'Gurukul': set(['Swaminarayana Gurukul']),
- 'Junction': set(['Prof Ashirvadam Junction', 'Townhall Junction']),
- 'Layout': set(['NGF Layout']),
- 'PALYA': set(['PAPAREDDY PALYA']),

- 'Palya': set(['Mariyappana Palya', 'Papareddy Palya']),
- 'Quarters': set(['Shirke KHB Quarters', 'University Quarters']),
- 'Road': set(['Escorts Yalahanka Road', 'Outer Ring Road']),
- 'Sarjapura': set(['Sarjapura']),
- 'Service': set(['Nexus Maruti Service']),
- 'Stop': set(['Vinayaka Layout Bus Stop']),
- 'Temple': set(['Ganesha Temple', 'Veeranjaneya Temple'])}

I removed the upper case and made all the types lower case to eliminate the duplicate entries.

Prof Ashirvadam Junction => prof ashirvadam junction

Townhall Junction => townhall junction

Nexus Maruti Service => nexus maruti service

Papareddy Palya => papareddy palya

Mariyappana Palya => mariyappana palya

Vinayaka Layout Bus Stop => vinayaka layout bus streetop

Mallathalli Cross => mallathalli cross

Shirke KHB Quarters => shirke khb quarters

Aladamara => aladamara

HAL Airport => hal airport

Nagarabhavi BDA Complex => nagarabhavi bda complex

Ambedkar College => ambedkar college

Nagarabhavi 9th Block => nagarabhavi 9th block

PAPAREDDY PALYA => papareddy palya

BMTC Depot-12 Gate => bmtc depot-12 gate

Lakshmisagar Gate => lakshmisagar gate

2. Secondly, I had to be very careful in creating the mapping as I had defined
   mapping = { #"St": "Street",
   #"St.": "Street",
   "Ave": "Avenue",
   "Rd.": "Road",
   "Circle":"Circle",

```
  "PALYA": "Palya"
}
```

Due to this, when there is a value of Bus stop, it used to get replaced with Bustreet stop

```
Vinayaka Layout Bus Stop => vinayaka layout bus streetop
```

Hence I had to remove the abbreviations used in mapping which required a careful study of data.

```
Vinayaka Layout Bus Stop => vinayaka layout bus stop
```

3. There are a lot of unique tag types in the osm file which requires the code to be adjusted accordingly every time. There is no standardization in the way the address or locality is defined.
4. There are cases where ambiguous names like “Cir”

   ```
   'Cir': set(['Aurobindo Cir']),
   ```

   Instead of “Circle” which then corrected as “Circle” for better clarity.
5. Abbrevations noted are expanded .

   ```
   'Complex': set(['Nagarabhavi BDA Complex']),
   ```

   ```
   Nagarabhavi BDA Complex => nagarabhavi bangalore development authority complex
   ```

6. There were house numbers with special characters which are corrected thereby removing the special characters.

7. There were local language usage in naming the tags.

   ```
   'Gurukul': set(['Swaminarayana Gurukul'])
   ```

Overview of the Data

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

File sizes

bengaluru_india.osm **89.2 MB**

bengaluru_india.osm.json **190.1 MB**

Code:

Types of tags

```
{'bounds': 1,
'member': 3,
'nd': 48,
'node': 739,
'osm': 1,
'relation': 1,
'tag': 182,
'way': 5}
```

Tag Keys analysis

Code:


```
{'lower': 168, 'lower_colon': 14, 'other': 0, 'problemchars': 0}
```

Unique User ids:

Code:


```
set(['123364',
    '1296080',
    '1306',
    '1319316',
    '136860',
    '1765920',
    '178915',
    '1829683',
    '183942',
    '20181',
    '2179',
    '2477516',
    '256444',
    '337433',
    '3516',
    '354670',
    '35811',
    '392516',
    '398086',
    '398735',
    '492742',
    '508',
    '586822',
    '587',
    '632616',
    '63375',
    '634020',
    '642345',
    '693794',
    '697874',
    '697960',
    '719005',
    '722137',
    '74061',
    '77582',
    '78656',
    '803459',
    '827808',
    '83660',
    '88870',
    '89411',
    '91490',
    '97701'])
```

The processed map has been saved to bengaluru_india_audit.osm.json .we have processed the

audited map file(as mentioned as first code above) into array of JSON, to put it into mongodb instance. This will take the map that we have been audited. First we load the script to insert the map

```
data = process_map(bengaluru_india.osm')
pprint.pprint(data[0:6])
```

```
[{'created': {'changeset': '16957521',
              'timestamp': '2013-07-15T08:10:50Z',
              'uid': '634020',
              'user': 'user_634020',
              'version': '4'},
  'id': '17327077',
  'pos': [12.9026964, 77.5949117],
  'type': 'node'},
 {'created': {'changeset': '18611831',
              'timestamp': '2013-10-30T05:16:40Z',
              'uid': '634020',
              'user': 'user_634020',
              'version': '32'},
  'id': '17327092',
  'pos': [12.9063367, 77.5950592],
  'type': 'node'},
 {'created': {'changeset': '18598983',
              'timestamp': '2013-10-29T11:01:32Z',
              'uid': '634020',
              'user': 'user_634020',
              'version': '32'},
  'id': '17327095',
  'pos': [12.910516, 77.5987265],
  'type': 'node'},
 {'created': {'changeset': '2446958',
```

```
          'timestamp': '2009-09-11T16:14:48Z',
          'uid': '1306',
          'user': 'PlaneMad',
          'version': '74'},
 'highway': 'traffic_signals',
 'id': '17327106',
 'name': 'Aurobindo Circle',
 'pos': [12.9171587, 77.5858225],
 'type': 'node'},
{'created': {'changeset': '833006',
          'timestamp': '2009-03-19T17:09:30Z',
          'uid': '35811',
          'user': 'Praveen',
          'version': '29'},
 'id': '17327139',
 'pos': [12.9349712, 77.624083],
 'type': 'node'},
{'created': {'changeset': '8054691',
          'timestamp': '2011-05-05T06:28:55Z',
          'uid': '1306',
          'user': 'PlaneMad',
          'version': '21'},
 'id': '17327141',
 'pos': [12.9384996, 77.62914],
 'type': 'node'}]
```

```
{u'created': {u'changeset': u'16957521', u'version': u'4', u'user': u'user_634020', u'timestamp': u'20
13-07-15T08:10:50Z', u'uid': u'634020'}, u'_id': ObjectId('5553887d18249360b6026c26'), u'type':
u'node', u'pos': [12.9026964, 77.5949117], u'id': u'17327077'}
```

Top 1 Contributing user

```
db.bangalore.aggregate([{
      "$group":{
         "_id":"$created.user",
         "count":{
            "$sum":1
         }
      }
   },{"$sort":{"count":-1}},
   {"$limit":1}])['result']
```

```
[{u'_id': u'docaneesh', u'count': 113770}
```

2 data that have palya

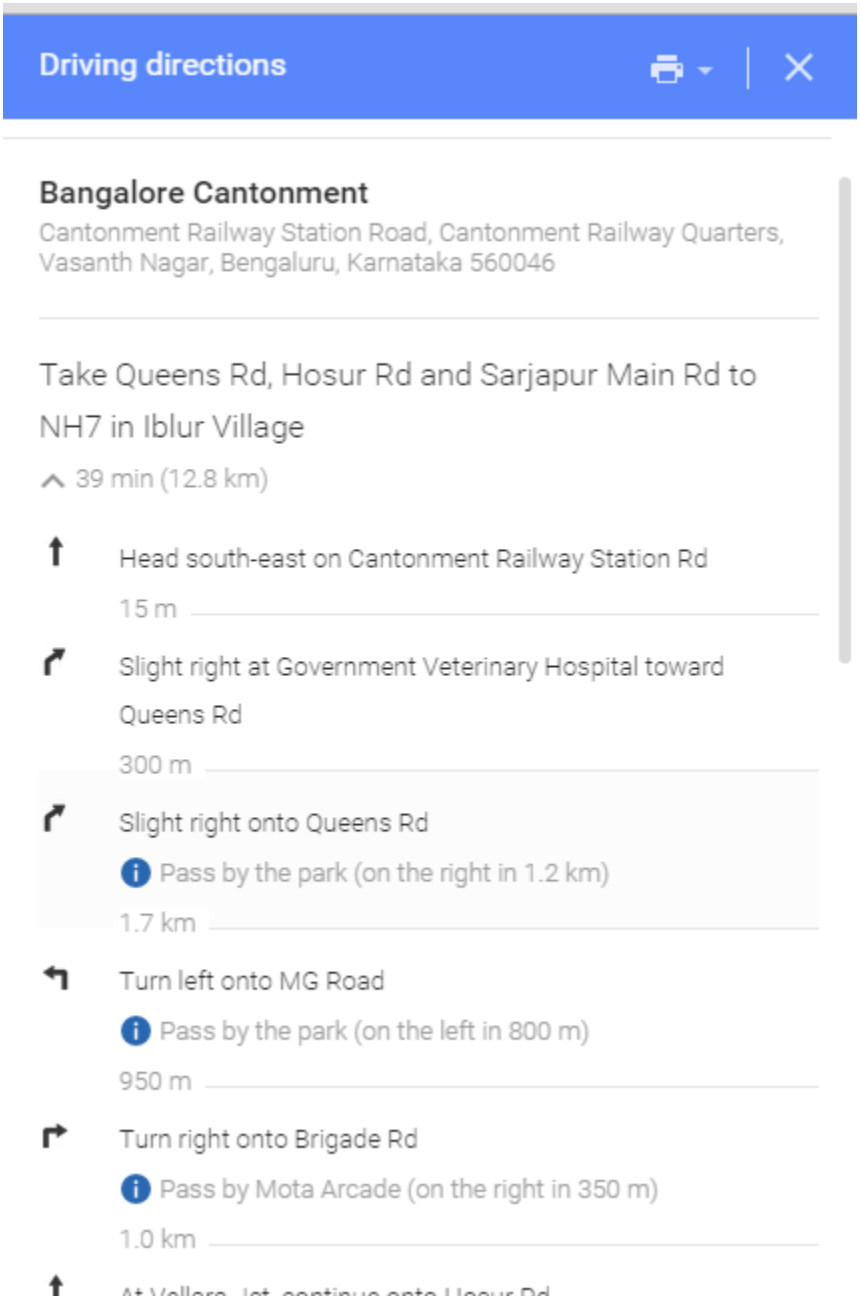
```
pipeline = [
        {'$match': {Palya:{'$exists':1}}},
        {'$limit' : 5}
]
result  = db.bengaluru_india.osm.aggregate(pipeline)['result']
pprint.pprint(result)
```

```
[{u'_id': u''Mariyappana Palya ', u'count': 1},
{u'_id': u''Papareddy Palya ', u'count': 1}
]
```

Other ideas about the datasets

Proposal to validate current dataset with external data:

1. Use google direction finder to validate if the OSM dataset matches with the details provided . Sample direction is given below.



OSM dataset can be corrected in case of any discrepancy to improve the accuracy.

2. Usage of apps like google now,zomato appto find near by attractions,restaurants and validate against the dataset.

Technical Difficulty: In gathering data form external tools.For this an api or webservice should be exposed to us.

Improve current dataset:

Create a data dictionary for mapping potential duplicate entries due to usage of abbreviations,wrong format,local language usage For eg : Gurukul= School;st=street.

Technical Difficulty: Identifying new entries and updating data dictionary at regular intervals.

Modify the business logic to validate the points against data dictionary.

Periodic review and correction of data:

If any of the existing points in the map is removed or changed, in situations like: If a shop is demolished or if a hotel is converted in to a saloon.

Technical Difficulty: Periodic physical inspection of address points to validate the content.