OpenStreetMap Sample Project
Data Wrangling with MongoDB

Deven Bhooshan

Map Area : Bengaluru(Bangalore), India

Problems in the map data

Postal Code

- It was found that Non numeric postal codes were present in the address. So these documents were ignored.
- Space separated codes were converted to correct 6 digit numeric numbers
  Eg : *560 090* was converted to 560090
- Documents having Postal Codes with length less than 6 were also ignored
  Eg : 79

Street Address

- At the time of processing the data, it was found that the city name *bangalore* was present at the end of many street addresses. So following regular expression was used to find and remove the presence of *bangalore* in the street address. Trailing spaces and commas were also removed.
  ```
  re.compile('[,\s]*bangalore[,\s]*$')
  ```
- Abbreviations were used in the street addresses. So the following transformation mapping was used to find and replace the presence of such words.
  ```
  { 'Rd.' : 'Road', 'Rd' : 'Road'}
  ```

Data overview **[Collection name in the database :** *tags***]**

File Size Description

bengaluru_india.osm : 604 MB
bengaluru_india.osm.json : 710 MB

- Total number of Documents
  ```
  db.tags.count()

  3467543
  ```
- Total number of nodes
  ```
  db.tags.find({'type': 'node'}).count()

  2818687
  ```
- Total number of unique users
  ```
  db.tags.distinct("created.user").length
  ```

```
1331
```

- Total number of *ways*

```
db.tags.find({'type': 'way'}).count()

648820
```

- Top contributing user

```
pipeline = [
        {'$group': {'_id': '$created.user', 'adds' : {'$sum':1}}},
        {'$sort' : {'adds' :  -1}},
        {'$limit': 1}
        ]
db.tags.aggregate(pipeline)

{ "_id" : "jasvinderkaur", "adds" : 126945 }
```

Some interesting insights

- 10 Amenities that are most common in 2 km radius of a hospital

```
[('cafe', 8834), ('school', 8902), ('bench', 9037), ('pharmacy', 9383), ('hospital', 9816),
('fast_food', 12198), ('place_of_worship', 18034), ('atm', 19757), ('bank', 19831),
('restaurant', 40654)]
```

  Most common is restaurant then bank and atm , place_of_worship , fast_food, hospital, pharmacy follow after that

- Average number of hospitals in ½ km radius of schools = 2 (python code below)

```
sum = 0
schools = db.tags.find({'amenity': 'school', 'pos': {'$exists': True}})
school_count = schools.count()
for school in schools:
        hospitals = db.tags.find({'amenity': 'hospital', 'pos' : {'$near': { '$geometry' : {
        'type': "Point" , 'coordinates' : school['pos']},  $maxDistance': 500}}})
         sum += hospitals.count()
print 'total hospitals', sum
print 'average', sum/school_count
```

- Only 3% of the documents have been contributed by the top contributor(`jasvinderkaur`)
- About 22% of the users contributed only 1 document.

```
pipeline = [
        {'$group': {'_id': '$created.user', 'adds' : {'$sum':1}}},
        {'$match': {'adds': 1}},
        {'$group': {'_id': null, count:{'$sum': 1}}}
        ]
db.tags.aggregate(pipeline)

{ "_id" : null, "count" : 293 }
```

- Total number of documents contributed by top 10 users is `1020667`. It is 30% of the total number of documents.

```
pipeline = [
        {'$group': {'_id': '$created.user', 'adds' : {'$sum':1}}},
        {'$sort' : {'adds' :  -1}},
        {'$limit': 10},
        {'$group': {'_id' : null, sum:{'$sum' : '$adds'} }}
        ]
db.tags.aggregate(pipeline)

{ "_id" : null, "sum" : 1020667 }
```

- *place_of_worships* and *hospital* count

```
pipeline = [
        {'$match': {'amenity': {'$in':['hospital', 'place_of_worship']}}},
        {'$group': {'_id': '$amenity', 'count':{'$sum' : 1} }}
        ]
db.tags.aggregate(pipeline)

{ "_id" : "place_of_worship", "count" : 817 }
{ "_id" : "hospital", "count" : 390 }
```

- Average number of docs per user

```
pipeline = [
        {'$group': {'_id': '$created.user', 'adds' : {'$sum':1}}},
        {'$group': {'_id' : null, avg :{'$avg': {'$sum' : '$adds'}}}}
        ]
db.tags.aggregate(pipeline)

{ "_id" : null, "avg" : 2605.2163786626597 }
```

- Top 5 reported/added amenities

```
pipeline = [
{'$match': {'amenity': {$exists:true}}},
{'$group': {'_id': '$amenity', count: {$sum:1}}},
{'$sort': {count:-1}}, {$limit: 5}
]
db.tags.aggregate(pipeline)

{ "_id" : "restaurant", "count" : 1139 }
{ "_id" : "place_of_worship", "count" : 817 }
{ "_id" : "atm", "count" : 601 }
{ "_id" : "school", "count" : 598 }
{ "_id" : "bank", "count" : 580 }
```

- Number of *ways* having more than 400 node *refs*

```
db.tags.find({'node_refs.400':{'$exists': true}}).count()
```

```
4
```

- Number of *ways* having more than 500 node *refs*

```
db.tags.find({'node_refs.500':{'$exists': true}}).count()

1
```

Other Ideas about the datasets

- Documents having name in *English Language* and name in local language (**Kannada)**

```
db.tags.find({'name': {'$exists': true}}).count()
25212

db.tags.find({'name:kn': {'$exists': true}, 'name': {'$exists': true}}).count()
7022
```

About only 28% of the documents have been reported/added to the openstreetmap library in the local language of Bengaluru.
And also you can see below only 31 documents have name in local language only

```
db.tags.find({'name:kn': {'$exists': true}, 'name': {'$exists': false}}).count()
31
```

Localization is very important for products like maps. Localizers not only translate the data but also brings local knowledge about the area. This local knowledge can greatly increase the overall value of openstreetmaps. The area which we have selected have only 28% documents in the local language. Organizations such as Mozilla and Google opened their 'l10n' platform to contributors. They created "Badges" and other ways to gamify the system. This gamification helped these organizations growing their 'l10n' community.

In general also, we can give incentives to users contributing significant amount. The incentives can be simple 'Badge' or 'Points'.

- The postal codes used in the openstreetmap data can be cross validated with some of other sources like http://www.mapsofindia.com/pincode/india/karnataka/bangalore/
Wide adoption of these sources by the community has made these sources less error prone so we can utilize these sources data for cleaning our openstreetmap data.

- Maps are more useful when it has listings of local businesses also. So there should be a separate platform for listing businesses in the openstreetmap as business listing requires some required attributes like address, phone number and other.

Conclusion

Bangalore City data is very large and has huge potential to drive awesome analytics. Even though the data has some human errors and lacks uniformity, but after cleaning the data, it can be used in any other projects. Things like localization and cross validating the data with other sources can greatly increase the overall user experience of openstreetmaps. The hypothesis that *Incentivizing the users with the 'Badges' and 'Point' system helps growing the overall community* has already been proved by the organizations like Mozilla. But this should be done in controlled way. Checks should be there to ensure that people are not abusing the platform.

The other sources like government websites can provide less error prone data. These websites has huge potential for cleaning and verification of the data entered by the contributors. For example if some contributor is adding some postal code for some locality, we can use the government websites to validate whether the postal code is correct for that particular locality or not. The only problem with this idea is to find such trustful resources. Many times it is very hard to find the resource which can be utilize for validation or cleaning purposes. And also a resource which looks trustful for me might not look same for other contributors.Quora is one website which solves this problem efficiently. Quora is a question - answer platform. Generally Quora displays most upvoted answers on top. So we can implement Gamification on data resources also. The localizers or contributors would upvote the most trustful resources. Most upvoted sources would be used for validating purposes.