Project: Wrangle OpenStreetMap Data

1.0 INTRODUCTION

AIM : To wrangle OpenStreetMap data for region of one's choice, using MongoDB, in a python environment.

Dataset Chosen : New-Delhi from Mapzen Metro Extracts

Link : https://mapzen.com/data/metro-extracts/metro/new-delhi_india/

Dataset size :	*Compressed XML: 44 MB* *Uncompressed XML: 718 MB*
Environment :	*Python v3.6.2* *MongoDB v3.4* *IDE: PyCharm*

2.0 WRANGLING

Issues Encountered Found During Auditing Phase

- **Abbreviated Street/Location Names** : There were many instances of locations and streets being abbreviated such 'St' for 'Street', 'Mkt' for 'Market', 'Extn' for 'Extension', 'Sec' for 'Sector' and so on. Apparently, hindi connotations for these words were found to be mostly correct, such 'Marg', 'Path', 'Sadak' etc..
- **Incorrect phone number format**: While divided into mobile and landline, the formats followed were highly inconsistent and all over the place. The standard national format in India is as follows:
    - Mobile : +91 XXXX NNNNNN
    - Landline: +91 11 XXXXYYYY (For Delhi only)
- **State Names :** It was found that the included areas from the surrounding regions as well. This included Gurgaon, Noida, Faridabad and Ghaziabad. Their state names were not proper such as Hryana (*Haryana*), U.P.(*Uttar Pradesh*) etc.
- **Sources :** Many sources such Bing, Yahoo, Survey etc. were either incorrectly spelled, or in a bad format.

- **Postal Code :** Delhi and surrounding areas postal code is of six digits only, and many such codes were encountered where the length was not six.

Cleaning the Dataset

- The mentioned street types were cleaned using **regex** functions, as it was not necessary that the word to be amended was the last word.
- Mobile number formats were corrected to a very high degree through **formatting and regex**, while landline number could not be done due to a large number of inconsistencies.
- Correct state names were stored in a **dictionary** against the wrong ones, and were corrected through it.
- An approach similar to the States was followed for sources as well.
- Postal codes not having length equal to six were assigned the value **NaN**.

3.0 DATA OVERVIEW

Initial dataset size (Uncompressed): *718 MB (delhi_map.osm)*

Converted JSON File size: *832 MB (delhi_map.osm.json)*

Database Queries

> Total number of records:

```
> db.delhi.find().count()

4110208
```

> Total number of “Node” entries:

```
> db.delhi.find({element_type:"node"}).count()

3415074
```

> Total number of “Way” entries:

```
> db.delhi.find({element_type:"way"}).count()

695134
```

> Total unique users:

```
> db.delhi.distinct("creation_info.uid").length

1498
```

> Total five contributors:

```
> db.delhi.aggregate([{$group:{"_id":"$creation_info.user",
count:{"$sum":1}}},{ $sort:{"count":-1}},{$limit:5}])

{ "_id" : "Oberaffe", "count" : 268668 }
{ "_id" : "premkumar", "count" : 164032 }
{ "_id" : "saikumar", "count" : 159906 }
{ "_id" : "Naresh08", "count" : 136335 }
{ "_id" : "anushap", "count" : 133391 }
```

> Average number of posts per user:

```
> db.delhi.aggregate([{$group:{"_id":"$creation_info.user",
count:{"$sum":1}}},{ $group:{"_id":"Average Posts Per User",
average:{"$avg":"$count"}}}])

{ "_id" : "Average Posts Per User", "average" : 2743.7970627503337
}
```

> Top five amenities:

```
> db.delhi.aggregate({$match:{"amenity":{$exists:1}}},
{$group:{"_id":"$amenity",count:{$sum:1}}},{$sort:
{"count":-1}},{$limit:5})

{ "_id" : "school", "count" : 954 }
{ "_id" : "place_of_worship", "count" : 407 }
{ "_id" : "parking", "count" : 353 }
{ "_id" : "restaurant", "count" : 249 }
{ "_id" : "fuel", "count" : 246 }
```

> Top five roadways:

```
> db.delhi.aggregate({$match:{"highway":{$exists:1}}},
{$group:{"_id":"$highway",count:{$sum:1}}},{$sort:
{"count":-1}},{$limit:5})

{ "_id" : "residential", "count" : 106059 }
{ "_id" : "service", "count" : 13178 }
{ "_id" : "living_street", "count" : 5958 }
{ "_id" : "tertiary", "count" : 4533 }
{ "_id" : "unclassified", "count" : 3117 }
```

> Top five data sources:

```
> db.delhi.aggregate({$match:{"source":{$exists:1}}},
{$group:{"_id":"$source",count:{$sum:1}}},{$sort:
{"count":-1}},{$limit:5})

{ "_id" : "Bing", "count" : 649 }
{ "_id" : "Yahoo", "count" : 315 }
{ "_id" : "Survey", "count" : 57 }
{ "_id" : "GPS", "count" : 39 }
{ "_id" : "AND", "count" : 39 }
```

> Top five areas, ranked using postal code

```
> db.delhi.aggregate({$match:{"addr.postcode":{$exists:1}}},
{$group:{"_id":"$addr.postcode",count:{$sum:1}}},{$sort:
{"count":-1}},{$limit:5})

{ "_id" : "110087", "count" : 509 }
{ "_id" : "122001", "count" : 100 }
{ "_id" : "110092", "count" : 65 }
{ "_id" : "100006", "count" : 59 }
{ "_id" : "110075", "count" : 55 }
```

4.0 <u>ADDITIONAL IDEAS</u>

- The data has a lot of inconsistencies in the address format, which either could be hardcoded in the cleaning file, during further cleaning, or can be corrected through online directories taken as standards.

- Postal Codes on the other hand, can be cleaned manually as well, because of a very high degree of format accuracy.

```
> db.delhi.aggregate({$match:{$and:[{"addr.postcode":
{$exists:1}},{"addr.postcode":"NaN"}]}},{$group:
{"_id":"$addr.postcode", count:{$sum:1}}})

{ "_id" : "NaN", "count" : 22 }
```

- While not being used presently, I think the phone number related fields can be cleaned using Python repositories available on Github for this purpose, such as this one(https://github.com/daviddrysdale/python-phonenumbers). This will help in cleaning of landline number as well, which are not being done presently.
- Finally, as this data is not limited to Delhi only and has some of the surrounding regions as well, it should be renamed to NCR-Map, signifying the ***National Capital Region.***

```
> db.delhi.distinct("addr.state")
[
        "Delhi",
        "NCR",
        "Uttar Pradesh",
        "Haryana"
]
```

It should be kept in mind, that the above suggested improvements are not absolute, and may lead to some issues during implementation. So as a best practise, and also a precautionary measure, it may prove worthwhile to anticipate such problems. These are documented as follows, as well as some additional ideas.

1. Due to future changes in street names or addresses, the present dataset may become obsolete. This will make cleaning harder as directories taken as standard will have updated data, and corresponding older names may be rendered invalid. One example of this is the city of Gurugram, which was previously named as Gurgaon. Gurugram is the official name, while Gurgaon is still being used in most unofficial documents.
2. Phone numbers can be formatted through the mentioned library, but as was seen in this dataset, a string containing multiple numbers with some digits replaced will be very difficult to correct. There is no standard for splitting such strings. One way that I think can be implemented to solve this Machine

learning. Since we do have a large dataset for PAN India, an ML algorithm may be trained, to at the least, identify the issue which can later be cleaned manually.

3. For a large portion of this data, it was observed the address data was generally inconsistent across fields, i.e. Street names also had house numbers, or house numbers included area identifiers, and such data could not be cleaned as there was no pattern. Hard-coding would be a waste of resources in this case. So, the solution stated before, of training an ML algorithm, could also be applied here.
4. As a lot of data seemed to be manually entered, it may pose a problem if this data isn't actually accurate especially in the case of address. And that may even lead to wrong cleaning methods being applied. So, after a *cleaned address* is built through whatever methods, the same can be validated using the location tags, as it seems to be highly accurate (more number of decimal places). The address can then be verified using latitude and longitude. A very good website http://www.latlong.net, can be used for comparison.

5.0 CONCLUSION

After three to four iterations of auditing and cleaning the data using several samples, it's clear that the data has been manually entered, as well as gathered through GPS.

Manually entered data, if not from a credible source, cannot be relied upon a 100%. So, a robust mechanism to verify the data is needed, alongside cleaning practises such as those demonstrated by me in this project. The cleaning done by me only represents a small fraction of the requirements for this dataset.

REFERENCES:

- *https://en.wikipedia.org/wiki/National_Capital_Region_(India)*
- *https://docs.mongodb.com/manual/*
- *https://streets.openalfa.in/*
- *https://www.openstreetmap.org/*
- *http://www.callingdelhi.com/delhi-landline-and-mobile-phone-numbers*
- *https://mapzen.com/data/metro-extracts/metro/new-delhi_india/*
- *https://stackoverflow.com/*

Author : Shrey Marwaha