Wrangle OpenStreetMap Data

Introduction:

- Map Area: Mumbai, Maharashtra, India
- Dataset: This data contains information about the city Mumbai also called the 'city of dreams' since the Indian Movie industry is located in this city also known as Bollywood. Every year thousands of people come here to either get in the movie industry or music industry. Besides all this there are many tourists spots and old monuments. I have chosen this dataset since i live in a city nearby called Pune. I wanted to explore what kind of information the Open Street Map data contains about Mumbai since this is an important place so i wanted to contribute to the data by removing any errors if any.

Problems Encountered in the Map

1. Identifying the problems in the extract

Size of the dataset : 406 MB Size of sample extract I have created by the code which was given: 41.1 MB

I used two techniques for identifying any problems in the sample dataset.

- Analyzing the sample osm data itself by opening in an editor.
- Modifying the "audit.py" and "process_osm.py" which created the sample osm JSON file and after wards analyzing that data set to search or identify errors if any.
- Again repeat the steps above.

2.Problems faced

After doing the analysis I found four main problems. These are:

a. Over-abbreviated Street Names and Street names in regional language.

For correcting this I mapped the inconsistent data to correct data. A sample example is shown below.

- "Vasai Station Rd" : "Vasai Station Road"
- "Yashavant Nagar Rd" : "Yashavant Nagar Road"
- "mankoli naka" : "mankoli Toll"

There were also inconsistent street name which made no sense at all and i was not sure what to do so i took them in a lsit and removed those nodes or ways from my analysis.

Invalid street names = ["govandi" , "World" , "compound" ,"Powai", "Multiplex", "Gokuldham"]

b. Incorrect Phone number format.

The data has several numbers which are in the wrong format, some of them are given below.

- '022 2643 5361'
- '+91 22 2265 4194 / 2263 0393'
- '+98 22 65 28 52 84'
- '8108957786'
- '(91)-9972526110'

First of all the length of a phone number should be 10 and it should be preceded by our country code '+91' for example

- "(91)-9972526110" to "+919972526110"
- "022 2643 5361" to "+912226435361"

c. Incorrect cuisine and bank names

After auditing the data and creating the JSON file. WHen i started to query the data. I found out that the names of these attributes are inconsistent. SO like street name data I mapped them to thier correct description. Below are some sample mappings I have done.

Cuisine Mapping

- "burger" : "American",
- "pizza" : "Italian",
- "coffee" : "Italian",
- "regional": "Maharashtrian"

Bank Names Mapping

- "Greater" : "Greater Bombay Co-operative Bank",
- "Union" : "Union Bank of India ",
- "State" : "State Bank of India",
- "SBI" : "State Bank of India",

d. Name field not there in some of the documents

While analyzing the data I found that some documents did not have the name field in it, instead their name was given in different fields like "operator" or "brand. So I modified the "process_osm.py" to get the name field populated in such cases.

Auditing the data and filtering out inconsistent information

For this purpose i have written the code below to implement the same.

1.sample_creater.py

Creates a sample extract of the main OSM file so that it is easy to work with the data.

2. audit.py

This code audits the data extract, finds the problems and updates them with proper information.

3. process_osm.py

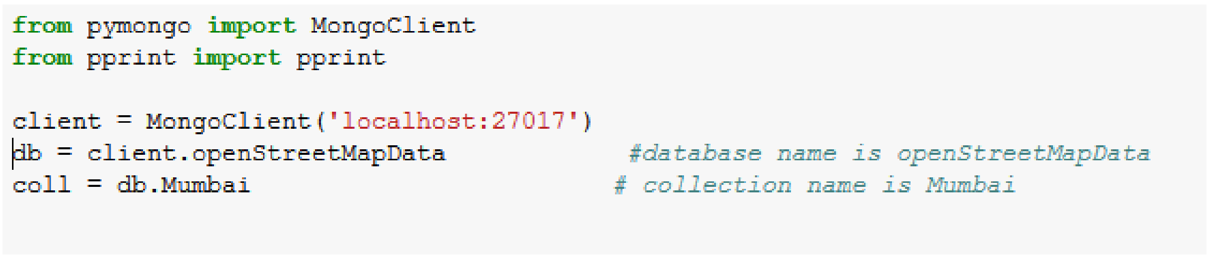
This file is very important since here I am creating the JSON type document for MongoDB plus it also does some small data cleaning also.

Importing data to MongoDB

For this purpose I am using the JSON file created above and running this command in the windows command prompt

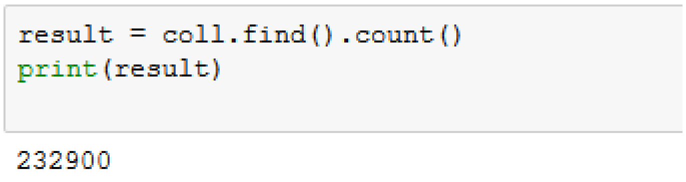
mongoimport --db openStreetMapData --collection Mumbai --type json --file sample_mumbai.osm.json

Connecting to MongoDB



```
from pymongo import MongoClient
from pprint import pprint

client = MongoClient('localhost:27017')
db = client.openStreetMapData                #database name is openStreetMapData
coll = db.Mumbai                            # collection name is Mumbai
```

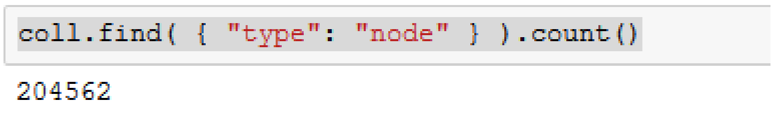
Number of Total Documents (nodes and ways)



```
result = coll.find().count()
print(result)
```

```
232900
```

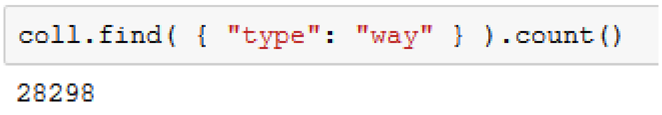
Total number of Nodes in the document



```
coll.find( { "type": "node" } ).count()
```

```
204562
```

Total number of Ways in the document



```
coll.find( { "type": "way" } ).count()
```

```
28298
```

Number of unique contributers

```
len(coll.distinct( "created.user" ))
```

```
873
```

User ID who has contributed the most

```
def pipeline():
    pipeline = [{"$match" : {"created.user" : {"$exists" : True , "$ne" : None }}},

                {"$group": {"_id" : "$created.user",
                            "count" : {"$sum" :1}}},

                {"$sort" : {"count" : -1}},

                {"$limit" : 1}

                ]

    return pipeline

output = pipeline()
result = coll.aggregate(output)
for i in result:
    pprint(i)
```

```
{'_id': 'PlaneMad', 'count': 7852}
```

List of top 10 amenities in the city

```
def pipeline():
    pipeline = [{"$match" : {"amenity" : {"$exists" : True , "$ne" : None }}},

                {"$group": {"_id" : "$amenity",
                            "count" : {"$sum" :1}}},

                {"$sort" : {"count" : -1}},

                {"$limit" : 10}

                ]

    return pipeline

output = pipeline()
result = coll.aggregate(output)
for i in result:
    pprint(i)
```

```
{'_id': 'place_of_worship', 'count': 53}
{'_id': 'restaurant', 'count': 41}
{'_id': 'bank', 'count': 30}
{'_id': 'school', 'count': 29}
{'_id': 'cafe', 'count': 19}
{'_id': 'hospital', 'count': 18}
{'_id': 'atm', 'count': 17}
{'_id': 'fast_food', 'count': 17}
{'_id': 'parking', 'count': 15}
{'_id': 'fuel', 'count': 14}
```

Since India is a country which has many religions there must be many places of worship for all those religions. So i guess the output is justifying it.

How many fuel stations are there in the city

```
coll.find( { "amenity" : "fuel" }).count()
```

```
14
```

Number of atms in sample location of Mumbai I have selected

```
result = coll.find({"atm" : "yes"}).count()
print(result)
```

```
36
```

Which bank has the most ATMs

```
def pipeline():
    pipeline = [{"$match" : {"name" : {"$exists" : True , "$ne" : None },
                            "atm" : "yes"}},

                {"$group": {"_id" : "$name",
                            "count" : {"$sum" :1}}},

                {"$project": {"name" : "$name",
                             "number_of_atms": "$count"}},

                {"$sort" : {"number_of_atms" : -1}}

                ]

    return pipeline

output = pipeline()
result = coll.aggregate(output)
for i in result:
    pprint(i)
```

```
{'_id': 'HDFC Bank', 'number_of_atms': 5}
{'_id': 'State Bank of India', 'number_of_atms': 5}
{'_id': 'Union Bank of India ', 'number_of_atms': 3}
{'_id': 'ICICI Bank', 'number_of_atms': 2}
{'_id': 'Kotak Mahindra Bank', 'number_of_atms': 2}
{'_id': 'Raheja Classique Bank', 'number_of_atms': 1}
{'_id': 'Karnataka Bank', 'number_of_atms': 1}
{'_id': 'private', 'number_of_atms': 1}
{'_id': 'Corporation Bank', 'number_of_atms': 1}
{'_id': 'THE HINDUSTHAN CO-OP.BANK LTD MUMBAI', 'number_of_atms': 1}
{'_id': 'Dena Bank', 'number_of_atms': 1}
{'_id': 'AXIS Bank', 'number_of_atms': 1}
{'_id': 'Punjab National Bank', 'number_of_atms': 1}
{'_id': 'IDBI', 'number_of_atms': 1}
{'_id': 'Bank of India', 'number_of_atms': 1}
{'_id': 'HSBC Bank', 'number_of_atms': 1}
{'_id': 'Greater Bombay Co-operative Bank', 'number_of_atms': 1}
{'_id': 'DBS Bank', 'number_of_atms': 1}
```

We can see that HDFC bank and State Bank of India has the most number of ATMs

Most popular bank in the region

```
def pipeline():
    pipeline = [{"$match" : {"amenity" :{ "$eq" : "bank", "$ne" : None},
                             "name" : {"$exists" : True , "$ne" : None }}},

                {"$group": {"_id" : "$name",

                            "count" : {"$sum" :1}}},

                {"$project": {"name" : "$name",
                              "popularity_score": "$count"}},

                {"$sort" : {"popularity_score" : -1}}

                ]

    return pipeline

output = pipeline()
result = coll.aggregate(output)
for i in result:
    pprint(i)
```

```
{'_id': 'State Bank of India', 'popularity_score': 4}
{'_id': 'HDFC Bank', 'popularity_score': 4}
{'_id': 'Kotak Mahindra Bank', 'popularity_score': 3}
{'_id': 'Union Bank of India ', 'popularity_score': 3}
{'_id': 'ICICI Bank', 'popularity_score': 2}
{'_id': 'Union Bank of India', 'popularity_score': 1}
{'_id': 'Karnataka Bank', 'popularity_score': 1}
{'_id': 'Greater Bombay Co-operative Bank', 'popularity_score': 1}
{'_id': 'THE HINDUSTHAN CO-OP.BANK LTD MUMBAI', 'popularity_score': 1}
{'_id': 'Dena Bank', 'popularity_score': 1}
{'_id': 'American Express Bank', 'popularity_score': 1}
{'_id': 'ANZ Grindlays', 'popularity_score': 1}
{'_id': 'Central Bank', 'popularity_score': 1}
{'_id': 'Bank of India', 'popularity_score': 1}
{'_id': 'ABN Amro', 'popularity_score': 1}
{'_id': 'Punjab National Bank', 'popularity_score': 1}
{'_id': 'Greater Bank', 'popularity_score': 1}
```

Here also I have got the result as expected most popular banks are HDFC Bank and State Bank of India

Listing out the popular cuisines

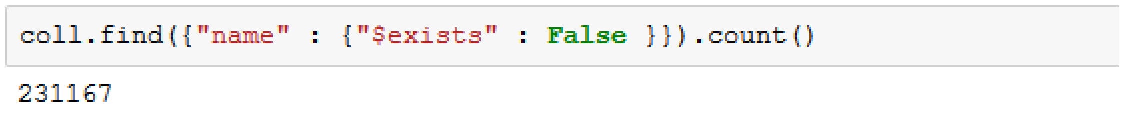
```
def pipeline():
    pipeline = [{"$match" : {"cuisine" :{ "$exists" : True, "$ne" : None}}},

                {"$project": { "cuisine": { "$toLower": "$cuisine" }}},


                {"$group": {"_id" : "$cuisine",

                            "count" : {"$sum" :1}}},


                {"$sort" : {"count" : -1}}

                ]

    return pipeline

output = pipeline()
result = coll.aggregate(output)
for i in result:
    pprint(i)
```

```
{'_id': 'indian', 'count': 15}
{'_id': 'italian', 'count': 11}
{'_id': 'maharashtrian', 'count': 4}
{'_id': 'american', 'count': 2}
{'_id': 'continental', 'count': 2}
{'_id': 'chinese', 'count': 2}
{'_id': 'seafood', 'count': 1}
```

Additional Improvements in Dataset

Earlier while querying the data, I saw there were numerous documents which did not have any name field , any address or any proper attribute which would highlight what the data is about. In MongoDB I have inserted those kind of documents, since it is not feasible to cleanse such kind of data. There can be thousands of such datasets where you don't have any proper highlighting key. So it be a tedious job to cleaning out all these uninformative elements. But we need to find a way to ignore these kind of data since these takes up much of the space. I hope to contribute to this by searching the internet and finding new ways to do this.

So now I find the number of documents that do not include name fields in them.



```
coll.find({"name" : {"$exists" : False }}).count()
```

```
231167
```

SO earlier we got total number of documents as 232900. SO we can see that only 1733 documents are perfectly formatted or we can say. About 99% of the data is filled with uninformative elements. So we need to find a way to remove them or they are taking up too many space. And this was just a sample what if there is a huge dataset. We need to find clever way to cleanse the data.

Conclusion

The Mumbai data set was very messy with too much of bad data. I tried to cover as much of data wrangling and data cleaning as possible for our analysis and for this project. And for Open Street Map I think they should modify or update their process of taking inputs such that when a user is updating any location it should be mandatory to give a key which defines the location or else we would get bad data out of it. I hope the analysis covers some interesting insights about the data set.

Bibliography

1. https://wiki.openstreetmap.org/wiki/OSM_XML
2. https://www.openstreetmap.org/#map=5/51.500/-0.100
3. https://mapzen.com/data/metro-extracts/
4. https://docs.mongodb.com/manual/
5. https://stackoverflow.com/¶