Udacity Data Analysis Project 3
OpenStreetMap Wrangling
Michael Collison

Map area: Seattle, WA, United States
As found on https://mapzen.com/data/metro-extracts

Associated Files

After downloading the OSM data from mapzen, I used convert_to_JSON.py to convert to JSON. I then created a new mongoDB database 'openstreetmap' and collection 'seattle'. I then read the JSON file into the seattle collection using the following terminal prompt.

```
mongoimport --db openstreetmap --collection seattle --file
seattle_washington.osm.json
```

The file mongo_query.py is used to make all queries found below. Update_phone.py and plot_user_contributions.py are discussed later when relevant.

Data Overview

File sizes:
seattle_washington.osm = 1.53 GB
seattle_washington.osm.json = 2.18 GB

Document Contents:

```
> db.seattle.find().count()
```
Number of documents = 7499839

```
> db.seattle.find({"type":"node"}).count()
```
Number of nodes = 6840942

```
> db.seattle.find({"type":"way"}).count()
```
Number of ways = 658746

```
> db.seattle.distinct({"created.user"}).length
```
Number of distinct users = 2623

```
> db.seattle.aggregate([{"$group":{"_id":"$created.user",
"count":{"$sum":1}}}, {"$group":{"_id":"$count",
"num_users":{"$sum":1}}}, {"$sort":{"_id":1}},
{"$limit":1}])
```
Number of users who posted once = 500

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

Top users # of posts = 1232388
User #50 = 20758
User #100 = 4569

Data Problems

An initial audit of the data revealed the data to be well cleaned. Three main problems were addressed; abbreviations of street address data, inconsistent phone number format, and invalid phone numbers.

Street Address Abbreviations:

There were a large number of variations in the abbreviations of street addresses. These include common abbreviations such as N. for North, St. for Street, etc. as well as a lack of capitalization. In the course of cleaning this field I opted for maximum explicitness, changing all common abbreviations to long-form and ensuring capitalization programmatically. I chose to clean the data while converting from XML to JSON in convert_to_JSON.py before adding to a mongoDB database. A full list of changes can be found in the aforementioned file in the mapping dictionary.

Phone number formatting:

There were a large number of variations in the way that phone numbers were formatted. A number of variations are shown below.

XXX-XXX-XXXX (desired)
X-XXX-XXX-XXX (also ok)
+XXXXXXXXXX
+X-XXX-XXX-XXX
(XXX) XXX-XXXX
+X XXX XXX XXXX
+X XXX XXX-XXXX
X XXX XXX XXXX
XXX.XXX.XXXX

I decided that the top variation (and the one below for numbers with a 1 in front) was the most effective format for phone number readability. I programmatically converted these by stripping all non-numerical characters and adding "-"s as appropriate.

Incorrect phone numbers:

After applying these changes, another audit showed ~50 remaining phone numbers either too long, too short, or empty. As these must be corrected on a per-location basis, I changed the "phone" tag to "FIXME", for correction in-browser.

Additional Ideas

Upkeep:

Aside from some cleaning in the addresses and phone numbers, the dataset for Seattle appears to be in good hands. A plot of the contributions of the top 50 users as created in plot_user_contributions.py shows an active community for OpenStreetMap in Seattle. With the cleanliness of the data not in question, auditing for accuracy and completeness is a logical next step to take in improving the dataset. What this would involve is a community effort to check and update all non-residential building tags to ensure they reflect current ownership over time. This is not a task that can be done programmatically, and so the presence of an active community is essential for this kind of upkeep.

Additional Data Analysis

```
db.seattle.aggregate([{"$match":{"man_made":{"$exists":1}}},
{"$group":{"_id":"$man_made", "count":{"$sum":1}}},
{"$sort":{"count":-1}}, {"$limit":10}])
```

Most common manmade feature: Pier

```
db.seattle.aggregate[{"$match":{"amenity":{"$exists":1}}},
{"$group":{"_id":"$amenity", "count":{"$sum":1}}}, {"$sort"
:{"count":-1}}, {"$limit":10}]
```

Most common amenities: Parking, bicycle parking, Restaurants

```
db.seattle.aggregate([{"$match":{"amenity":{"$exists":1},
"amenity":"restaurant"}}, {"$group":{"_id":"$cuisine",
"count":{"$sum":1}}}, {"$sort":{"count":-1}}, {"$limit":10}])
```

Most common cuisines: Mexican, Pizza

References

1. Banbury, Matthew. OpenStreetMap Sample Project. https://docs.google.com/document/d/1F0Vs14oNEs2idFJR3C_OPxwS6L0HPliOii-QpbmrMo4/pub?embedded=True
2. https://wiki.openstreetmap.org/wiki/List_of_OSM-based_services
3. https://docs.mongodb.org/manual/