OpenStreetMap Project
Data Wrangling with MongoDB

Name: Graeme Hay
Email: grmhay@gmail.com

Map area of study: Phoenix, AZ, USA

Mapzen URL for data source: https://s3.amazonaws.com/metro-extracts.mapzen.com/phoenix_arizona.osm.bz2

Openstreetmap URL for data source:
http://www.openstreetmap.org/relation/111257#map=10/33.6060/-112.1258

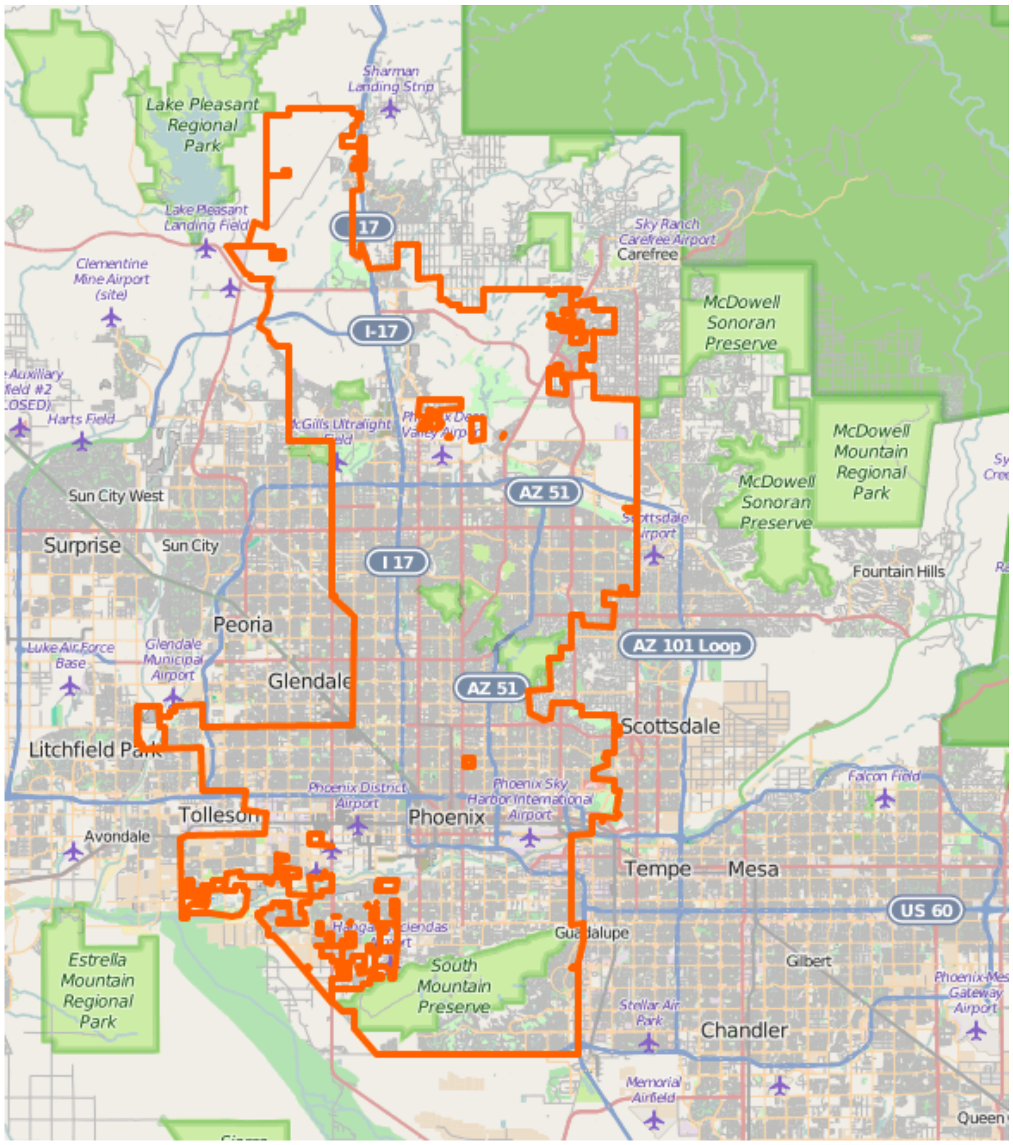

Figure 1: Openstreetmap rendering of Phoenix area map boundary

1. Problems Encountered in the Map Data

Using tagparser.py to generate a list of keys stored in each tags (top 30 shown below from output), this showed a somewhat unexpected frequency of occurrence of key values.

('highway', 249753), ('name', 126503), ('tiger:county', 80776), ('tiger:reviewed', 73866), ('tiger:name_base', 72544), ('tiger:cfcc', 71661), ('oneway', 70190), ('tiger:name_type', 67071), ('tiger:name_direction_prefix', 64734), ('tiger:zip_left', 60774), ('tiger:zip_right', 59098), ('surface', 53707), ('building', 45726), ('service', 41524), ('tiger:source', 33934), ('tiger:tlid', 33837), ('lanes', 32573), ('tiger:separated', 32366), (**'addr:city'**, 28387), ('power', 20296), (**'addr:state'**, 14998), ('addr:country', 14750), ('barrier', 14645), ('maxspeed', 13767) ('natural', 13718), ('bicycle', 13681), ('access', 11320), ('crossing', 11302), ('amenity', 10825), (**'addr:street'**, 10595)

It should be noted that 300 of the 636 key names that the parser returned had 10 or less occurrences in the OSM XML input file, thus we can conclude that there were many 'one-off' uses of tag key values by those submitting data. These should be ignored because of their lack of frequency of use and the danger of semantic equivalence between different low frequency key values (no dictionary has been agreed by the data submitters) – examples 'service_times', 'openhours', 'hours', 'opening_hours' look semantically equivalent.

Validity / Accuracy

From the Udacity course notes, a good test of validity uses key values that occurs frequently enough in the dataset to be a meaningful test of the data. They also need to be compared to some external source or measure to reflect upon the validity of this particular OSM data set being for the Phoenix AZ area.

Check: Telephone numbers. Looking for non-602/623 area codes in telephone numbers would be a good indicator of data set validity one would think, but only 675 key values are phone numbers in the overall data set. Comparing this frequency of occurrence to the other address key values like 'street' or 'city' as shown by tagparser.py output above, it can be concluded that this would be a poor test of validity.

Check: How many addresses have postal codes? How many of those addresses with postal codes have postal codes within the set of codes associated with Phoenix area?

How many addresses have postal codes? 8291 documents of the 2.25mm have the addr:postcode field defined. However, an examination of the tag key names in tagparser.py showed that many like key names to addr:postcode existed in the OSM

XML file that needed to be included – addr:postcode (8291), tiger:zip_left (60774), tiger:zip_left_1 (2782).

When the addr:postcode was analyzed in isolation for the zipcode distribution, it contained an outsized number of 85028 zip codes which is a particular area in central Phoenix. Examination of the XML file points to a single user 'Dr Kludge' who contributed these data points and perhaps a particular interest in mapping that area – possibly because the user lived there? However, when the tiger fields from the US Govt Census were added, the distribution of zip codes that could be uploaded into the Mongo DB by the parser appeared to be much more even.

The number of documents in Mongo after parsing and uploading with the zip code field populated was then 66,873.

```
db.phoenix.find ({"address.postcode": {"$exists" :1}}).count()
```

44,945 have postcodes outside 850** (Phoenix zip code) which would tend to lead to the conclusion that map data that belongs to adjoining cities has crept into the Phoenix data, which would negate the validity of the data set. Looking at Figure 1 above, the cities of Mesa, Glendale and Gilbert are clearly shown outside the box that is rendered on the map indicating the boundary of the Phoenix data set.

```
db.phoenix.find( {"address.postcode" : {"$regex" : /^85[1-9]/}},
{"id":1,"_id":0}).count()
```

Further running of the above query against the regex below shows the number of invalid zip codes present in the data for Phoenix.

851** - 251 (Apache Junction)
852** - 30,069 (Mesa, Peoria, Gilbert has 852)
853** - 14,622 (Glendale)
854** - 859** - 0

Completeness

Check: Compare the dataset to an external source (Census)

A quick check of census stats for the Phoenix area (http://quickfacts.census.gov/qfd/states/04/0455000.html) indicates there were 590,149 housing units and 112,202 firms in Phoenix in data that ranged in age from 2007 to 2010. However, this is sufficient enough to show that the number of documents that should have postal codes (>700k) vs those in OSM data (66k) would indicate the Phoenix data incomplete.

Consistency

Using tagparser.py, the number of elements of each element type was generated from the OSM data in order to look for inconsistent or non-standard use of element types.

The tags 'node', 'way' and 'relation' were expected from the Udacity course briefing and OSM wiki. The other sizeable element type was 'nd' which is used as a cross-reference to a node id in 'way' elements to create links between 'way points' for highways, roads, rivers etc. Given the focus of study for this exercise is nodes and ways, these could be ignored and parsed out before loading into the Mongo database.

No inconsistencies with the OSM XML specification for element types were found.

(partial output of mapparser.py)

```
{'bounds': 1, 'member': 11802, 'nd': 2693431, 'node': 2225705, 'osm': 1,
'relation': 1485, 'tag': 1684862, 'way': 278362}
```

Street name usage as a test of data consistency didn't appear to present an issue. Of the 4989 unique street names in the full data set, 41 street types did not match the expected list of 'typical' street names either because they were abbreviated or because of the 'regional' nature of the street name (e.g. North Camino del Sol). 205 unique street names therefore needed to be 'cleaned' which is <1% of the total data set, thus the observation that the data set was reasonably consistent.

The expected list used was
["Street","Avenue","Boulevard","Drive","Court","Place","Lane","Way","Trail","Road","Parkway","Highway","Freeway","Circle"]

Certainly, it was straightforward using the approach of a 'mapping' array (below) to clean up the truly inconsistent data in MongoImport.py:

```
Mapping = { "St": "Street",
            "St.": "Street",
            "Ave": "Avenue",
            "Rd.": "Road",
            "Rd" : "Road",
            "Blvd": "Boulevard",
            "Parkwway" : "Parkway"
          }
```

Conclusion is the capture method for the data looked of high quality.

2. Statistics on the Dataset

Size of the file

678MB JSON file after cleaning, 524MB XML file.

2504067 documents imported into MongoDB.

Number of unique users contributing data:

909 users contributed data.

```
db.phoenix.distinct("created.user").length
```

Top 5 contributing users contributed more than 61% of the OSM data for Phoenix – problem? (Distributed contribution should be power of the platform, no?)

Listing contributing users in descending order of contribution volume
```
{u'_id': u'Dr Kludge', u'count': 987419}
{u'_id': u'$user', u'count': 192218}
{u'_id': u'TheDutchMan13', u'count': 162143}
{u'_id': u'tomthepom', u'count': 106926}
{u'_id': u'adenium', u'count': 98144}
```

```
db.phoenix.aggregate([
                  {"$group" : {    "_id" : "$created.user",
                                             "count" : {"$sum" : 1 }}},
                  {"$sort"  :   { "count" : -1 }}
```

165 of the users contributed only once

```
db.phoenix.aggregate([{
... ...         '$group': {
... ...             '_id': '$created.user',
... ...             'count': {
... ...                 '$sum': 1
... ...             }
... ...         }
... ...     }, {
... ...         '$group': {
... ...             '_id': '$count',
... ...             'num_users': {
... ...                 '$sum': 1
... ...             }
... ...         }
... ...       }, {
... ...          '$sort': {
... ...             '_id': 1
... ...          }
... ...       }, {
... ...          '$limit': 1
... ...       }])
```

Number of nodes and ways

- 4989 unique street names

  `(from MongoImport.py)`

- 278362 ways

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

- 2225705 nodes

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

Number of chosen types of nodes like cafes, shops etc -

Using nodetagparser.py, a number of interesting node types were identified in the Phoenix dataset to incorporate into the MongoImport.py code. A sample showing the top occurring node types from the Phoenix OSM sample dataset (distribution of frequencies was the same in the full dataset) is shown below:

('**highway**', 6054), ('**power**', 1914), ('**crossing**', 1137), ('**natural**', 1021), ('**bicycle**', 783), ('**horse**', 777), ('**supervised'**, 763), (**'traffic_calming**', 759), ('**name**', 752), ('created_by', 651), ('addr:city', 588), ('**amenity**', 558), ('addr:street', 517), ('addr:housenumber', 490), ('addr:state', 435), ('addr:country', 413), ('barrier', 354), ('ele', 312), ('addr:postcode', 287), ('gnis:feature_id', 259), ('gnis:created', 219), ('railway', 214), ('source', 202), ('gnis:county_id', 198), ('gnis:state_id', 198), ('leisure', 146), ('entrance', 130), ('shop', 127), ……

The chosen ones to add to the Mongo database are highlighted in bold in the result set above:

- Highway appears to be used to indicate nodes like turning circles.
- Power appears to be types of power distribution apparatus – towers, substations etc.
- Crossing appears to be types of highway crossing (railway, unmarked pedestrian) etc

- Natural appears to be used in conjunction with describing mountain peaks as nodes.
- Bicycle and horse appear to be used with crossing node types to indicate the suitability for the crossing to be used for cyclists and horse riders respectively. Supervised is used to indicate whether the crossing is supervised or not by an attendant.
- Traffic calming is used to indicate the type of measure to used to control speed on a road (e.g. humps)
- Name and amenity are both used to describe the name of a node (e.g. a business) and the type of the business (e.g. bank).

Interestingly, it appears that the dataset is richer in node types that describe outdoor elements to do with leisure and travel than commerce. I included these node types as they are reflective of the outdoors orientation of Arizonans. I suspect the bias of the dataset contributors may also be towards outdoor pursuits.

3. Ideas to Improve Dataset

Eliminate the inconsistency is use of semantically-equivalent terms (open _hours, servicetimes, hours) from section 1.

Could have cleaned the zip code data to remove some inconsistent formatting of zip codes (e.g. 'AZ 85212'), but non 'xxxxx' format zip codes represented less than 1% of the data set. Given the question was to validate accuracy of the data by looking to ensure zip codes were in the Phoenix zip code range, this was not necessary at this time.