Geo sampling Documentation

Release 0.1.0

Suriyan Laohaprapanon, Gaurav Sood

October 19, 2016

Contents

Contents:

CHAPTER 1

Installation

1.1 Prerequisites

There are a couple dependencies that need to be built from the source on Windows so you may need to install Microsoft Visual C++ Compiler for Python 2.7.

1.2 Installation

Prepare the working directory. We recommend that you install in the Python virtual environment.

```
mkdir geo_sampling
cd geo_sampling
virtualenv -p python2.7 venv
. venv/bin/activate
```

Upgrade Python packages *pip* and *setuptools* to the latest version.

```
pip install --upgrade pip setuptools
```

Install geo-sampling package from test PyPI.

```
pip install --extra-index-url https://testpypi.python.org/pypi geo-sampling
```

CHAPTER 2

Usage

2.1 geo_roads

Get all the roads in a specific region from OpenStreetMap.

```
usage: geo_roads.py [-h] [-c COUNTRY] [-l {1,2,3,4}] [-n NAME]
                    [-t TYPES [TYPES ...]] [-o OUTPUT] [-d DISTANCE]
                    [--no-header] [--plot]

Geo roads data

optional arguments:
  -h, --help            show this help message and exit
  -c COUNTRY, --country COUNTRY
                        Select country
  -l {1,2,3,4}, --level {1,2,3,4}
                        Select administrative level
  -n NAME, --name NAME  Select region name
  -t TYPES [TYPES ...], --types TYPES [TYPES ...]
                        Select road types (list)
  -o OUTPUT, --output OUTPUT
                        Output file name
  -d DISTANCE, --distance DISTANCE
                        Distance in meters to split
  --no-header           Output without header at the first row
  --plot                Plot the output
```

2.1.1 Output File Format

1. *segment_id* - Unique ID (record number)
2. *osm_id* - ID from Open Street Map data
3. *osm_name* - Name from Open Street Map data (road name)
4. *osm_type* - Type from Open Street Map data (road type)
5. *start_lat* and *start_long* - Line segment start position (lat/long)
6. *end_lat* and *end_long* - Line segment end position (lat/long)

2.1.2 Examples

To get a list of all the country names:

```
geo_roads
```

To get a list of all boundary names of Thailand at a specific administrative level:

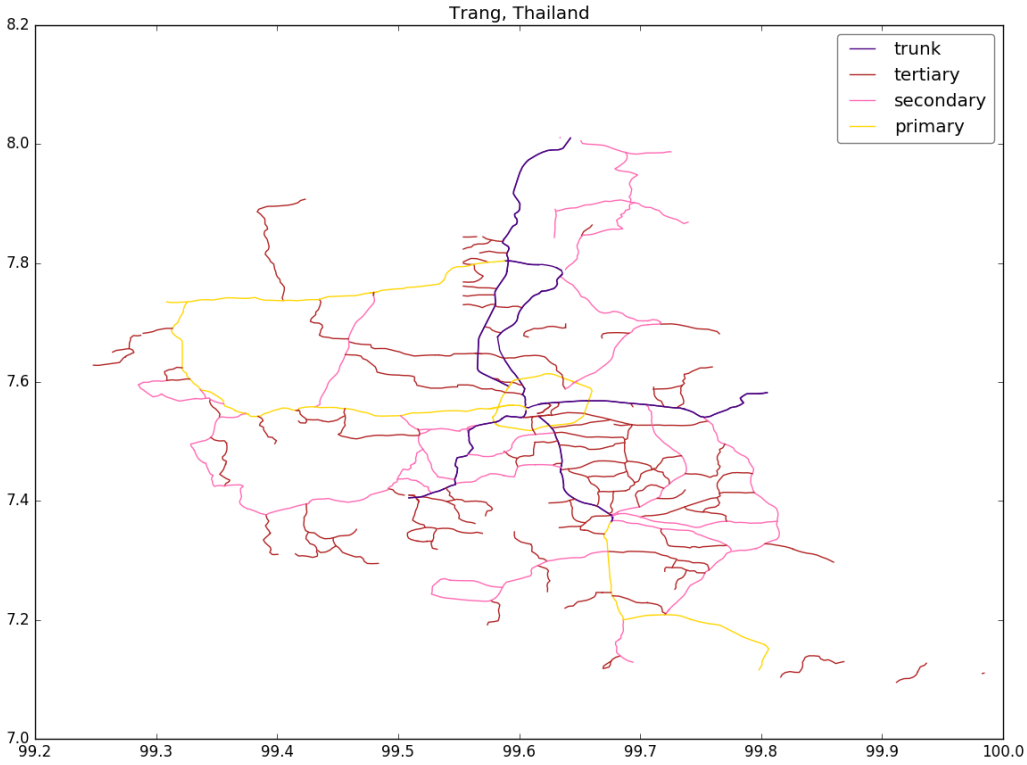
```
geo_roads -c Thailand -l 1
```

In this case, all boundary names (77 provinces) at the 1st administrative divisions level of Thailand will be listed.

To get road data for the `Trang` province (only the road types *trunk*, *primary*, *secondary* and *tertiary*):

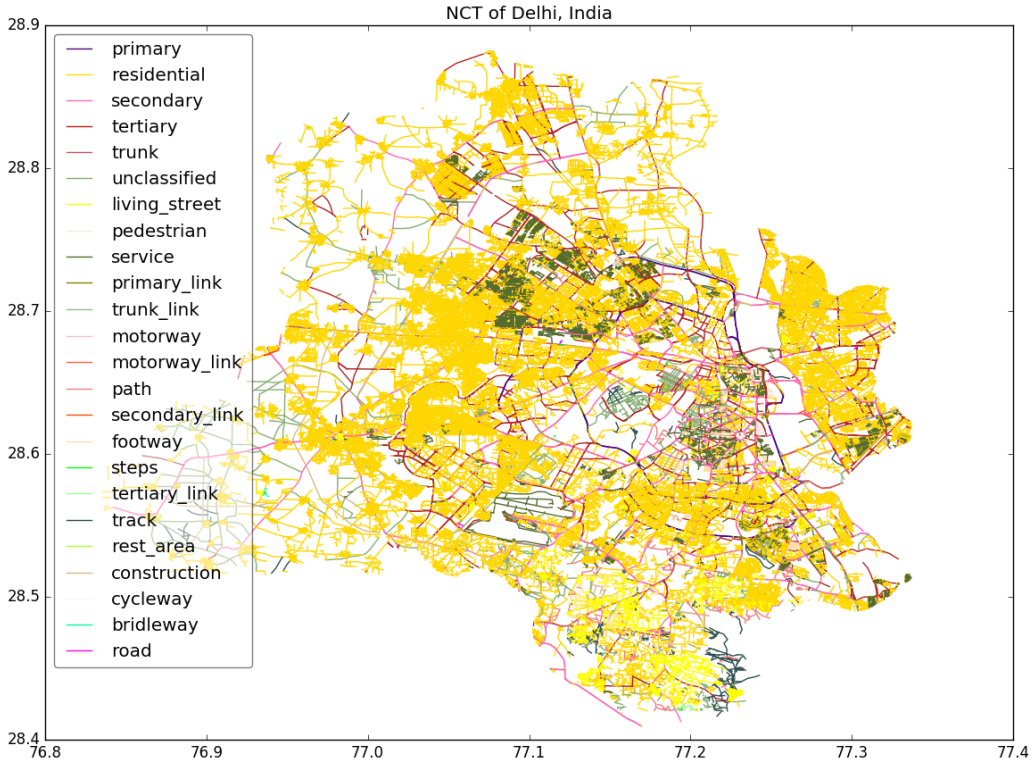
```
geo_roads -c Thailand -l 1 -n Trang -t trunk primary secondary tertiary --plot
```

Default output file will be saved as `output.csv` and all the road segments will be plotted if *–plot* is specified



To run the script for `Delhi of India` and to save the output as `delhi-roads.csv`:

```
geo_roads -c India -l 1 -n "NCT of Delhi" -o delhi-roads.csv --plot
```



By default, all road types will be outputted if *–types, -t* is not specified.

2.2 sample_roads

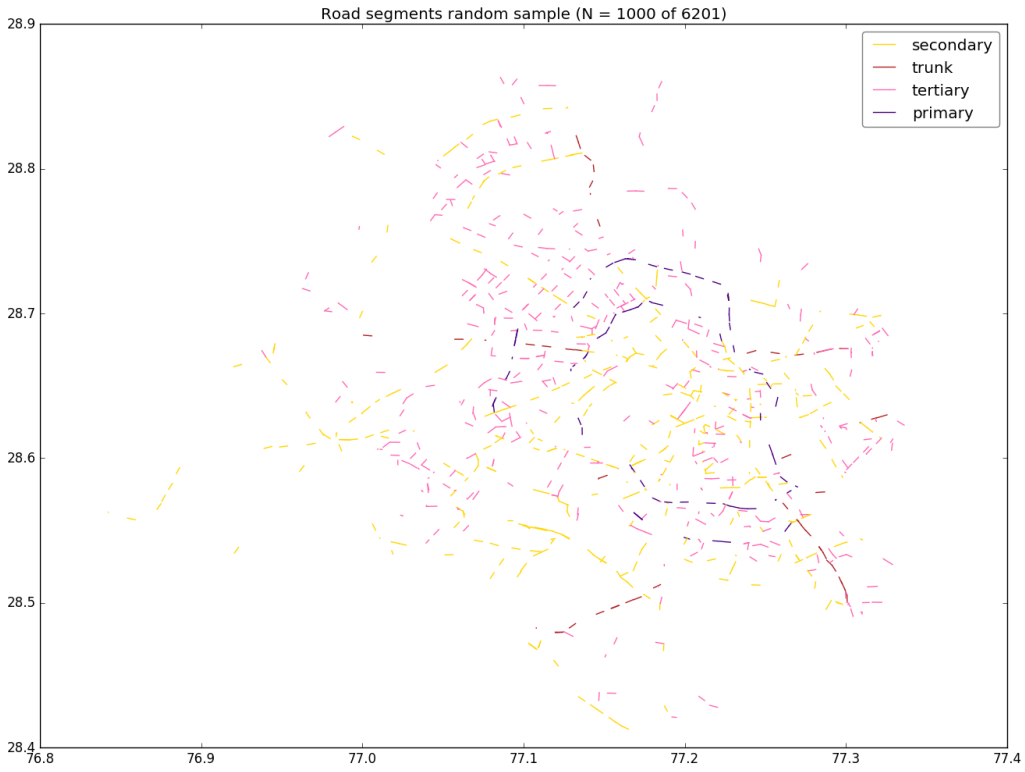
Randomly sample a specific number of road segments of all roads or specific road types.

```
usage: sample_roads.py [-h] [-n SAMPLES] [-t TYPES [TYPES ...]]
                           [-o OUTPUT] [--no-header] [--plot]
                           input

Random sample road segments

positional arguments:
  input                 Road segments input file

optional arguments:
  -h, --help            show this help message and exit
  -n SAMPLES, --n-samples SAMPLES
                        Number of random samples
  -t TYPES [TYPES ...], --types TYPES [TYPES ...]
                        Select road types (list)
  -o OUTPUT, --output OUTPUT
                        Sample output file name
  --no-header           Output without header at the first row
  --plot                Plot the output
```

2.2.1 Examples

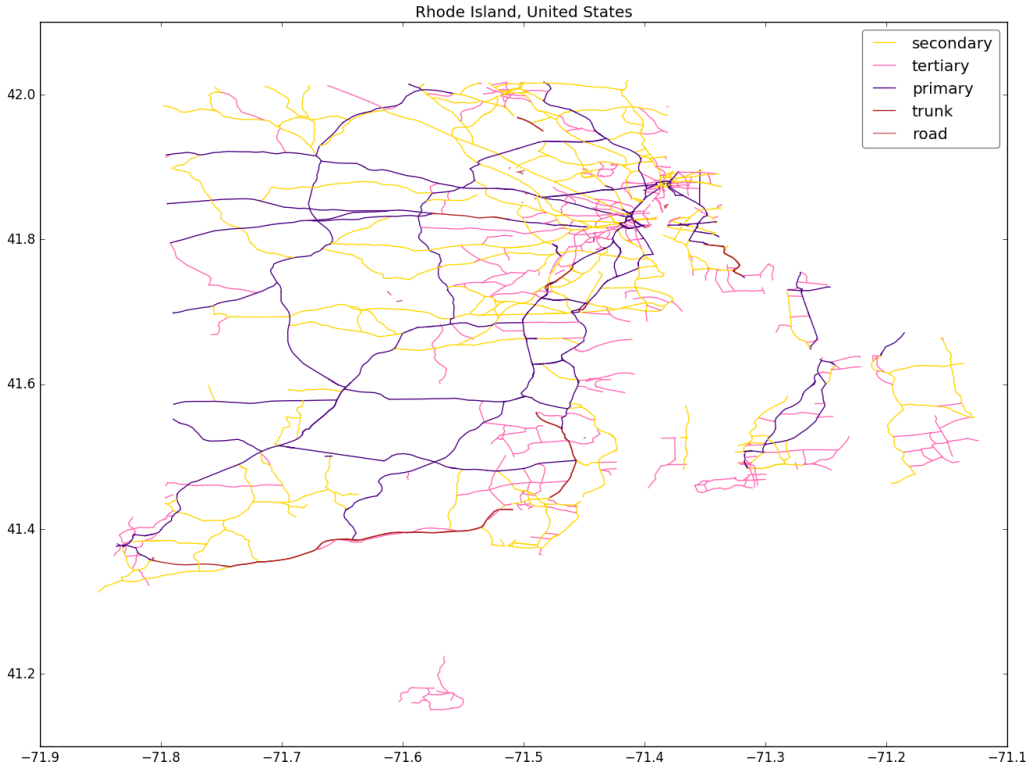
To get a random sample of 1,0000 road segments of road types *primary*, *secondary*, *tertiary* and *trunk*:

```
sample_roads -n 1000 -t primary secondary tertiary trunk -o delhi-roads-s1000.csv delhi-roads.csv
```



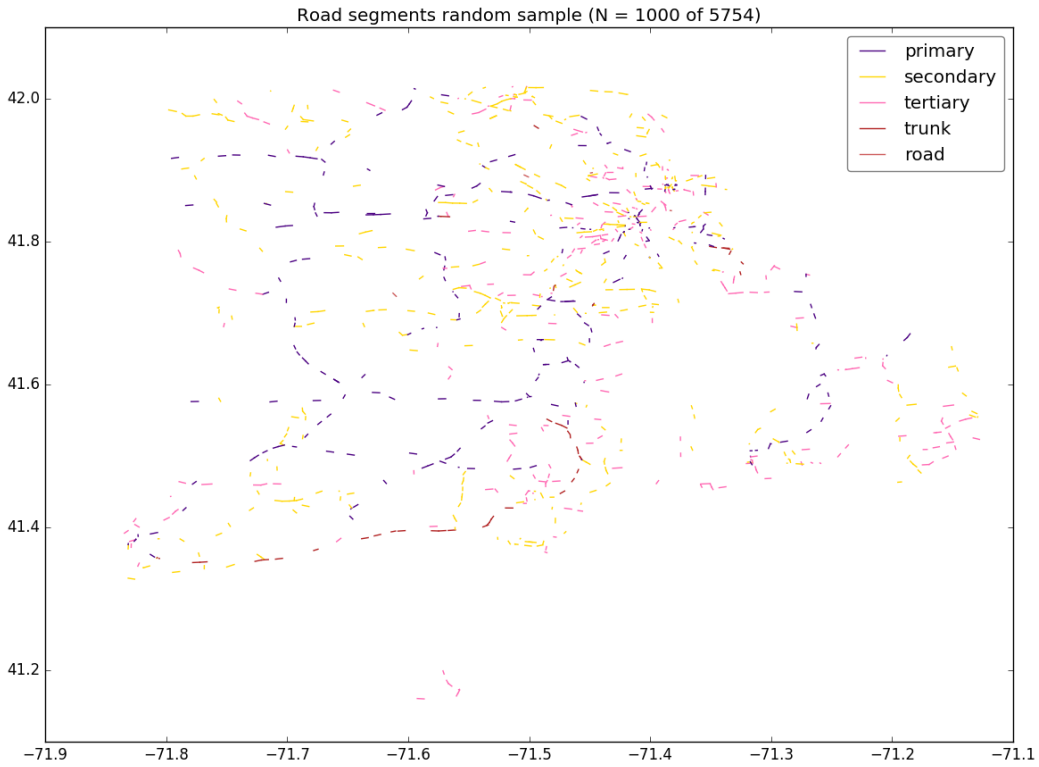
To get specific road types for Rhode Island in US:

```
geo_roads -c "United States" -l 1 -n "Rhode Island" -t trunk primary secondary tertiary road -o rhode
```



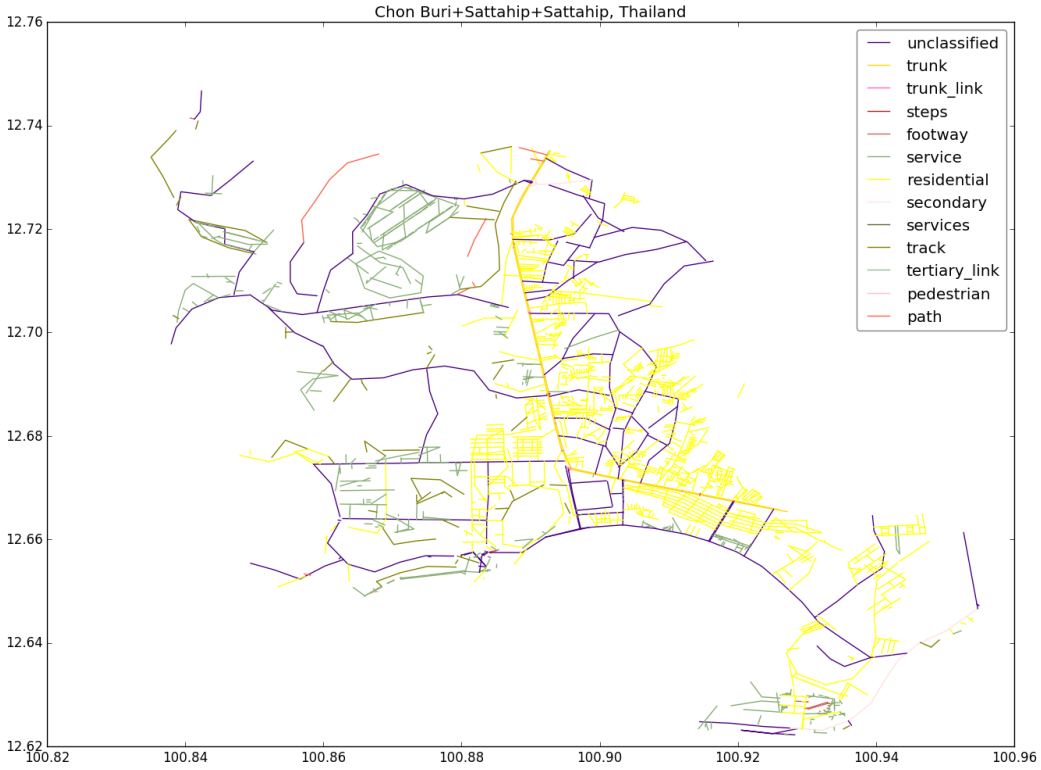
And then get a random sample of 1,000:

```
sample_roads -n 1000 -o rhode-island-s1000.csv --plot rhode-island-roads.csv
```



To get a specific region at 3rd adm. level (Tambon) of Thailand (e.g. "Tambon Sattahip, Amphoe Sattahip, Chon Buri, Thailand"):

```
geo_roads -c Thailand -l 3 -n "Chon Buri+Sattahip+Sattahip" -o sattahip-roads.csv --plot
```


Chon Buri+Sattahip+Sattahip, Thailand
unclassified
trunk
trunk_link
steps
footway
service
residential
secondary
services
track
tertiary_link
pedestrian
path
12.76
12.74
12.72
12.70
12.68
12.66
12.64
12.62
100.82
100.84
100.86
100.88
100.90
100.92
100.94
100.96

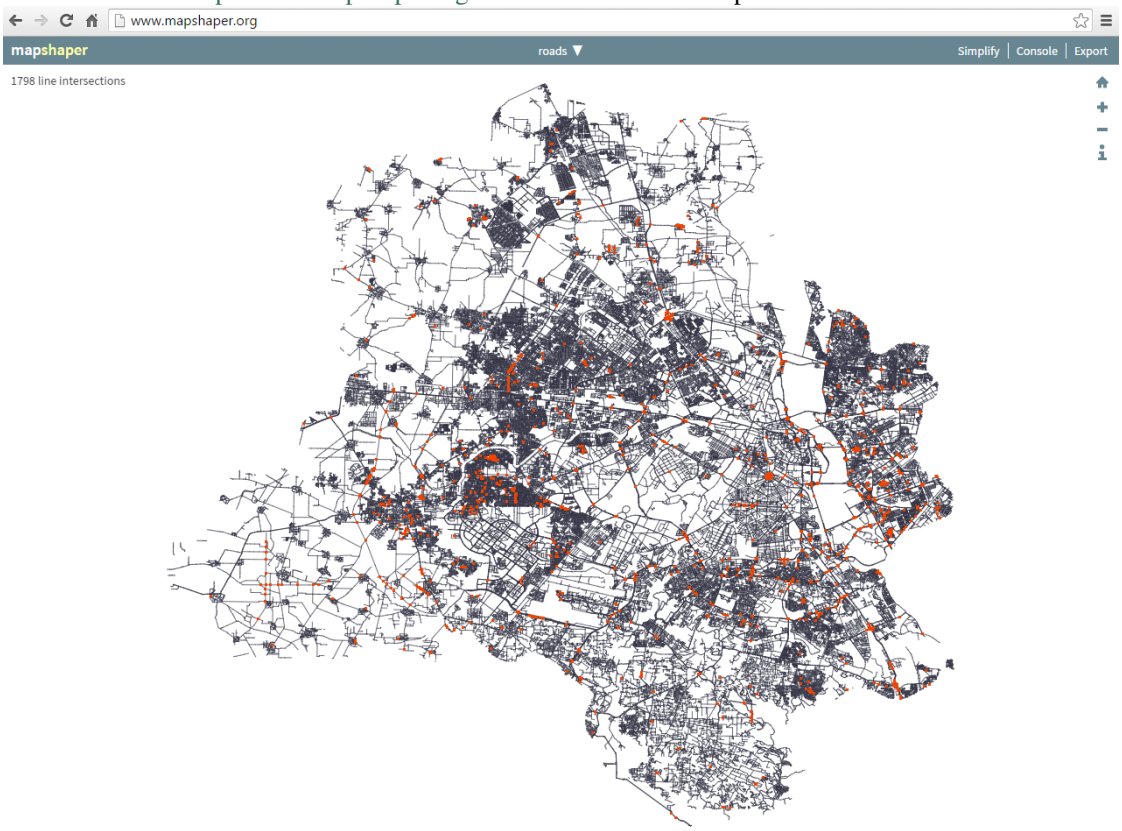
CHAPTER 3

Workflow

1. Start by downloading the administrative boundary data for the country in ESRI format from http://www.gadm.org/ country. For more information about administrative divisions of different countries, see https://en.wikipedia.org/wiki/Table_of_administrative_divisions_by_country There are multiple administrative levels — cities may be nested in states which may be nested in countries.
2. Using pyshp package load 2nd level shapefile (IND_adm2.dbf and IND_adm2.shp), extract polylines of "NCT of Delhi" and build map data extract URL for http://extract.bbbike.org like the one for .

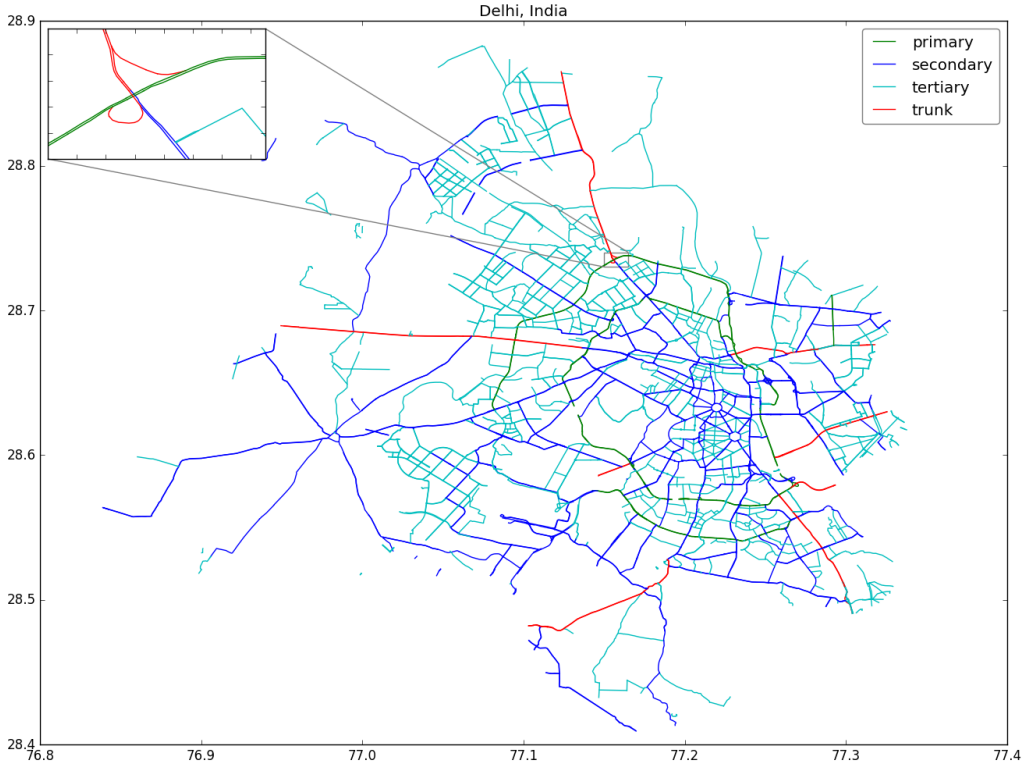
   This is so that we don't need to set up our own OSM map server, which is extremely large.

   Get link to the extracted map data by e-mail or check the download status page: http://download.bbbike.org/osm/extract/
3. Download and unzip it. There is a shapefile for road data in roads.*. Optionally we can drag and drop roads.* to view on http://www.mapshaper.org/. You'll see all roads map like this:

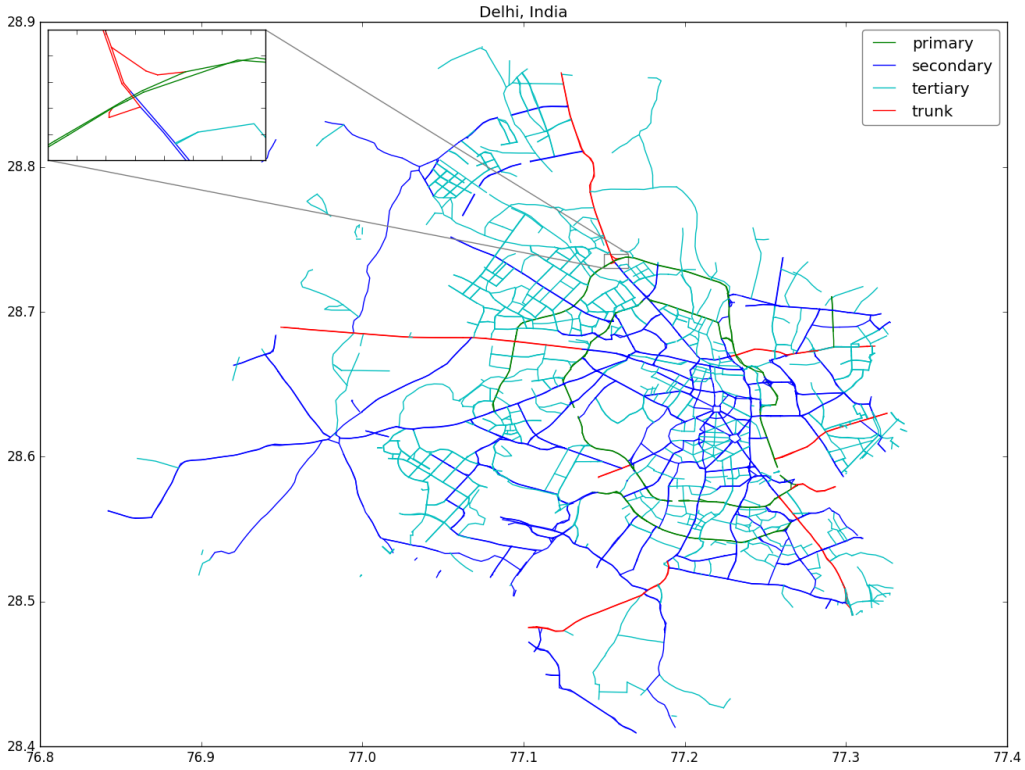


There are many types of roads found in the map data: 'primary', 'pedestrian', 'bridleway', 'secondary_link', 'tertiary', 'primary_link', 'service', 'residential', 'motorway_link', 'cycleway', 'secondary', 'living_street', 'track', 'motorway', 'construction', 'tertiary_link', 'trunk', 'path', 'trunk_link', 'rest_area', 'footway', 'unclassified', 'steps', and 'road'

4. Filter a few interesting road types and plot with matplotlib:



5. Iterate through all selected road types and split the polyline into 500 meters segments. The following figure plots segmented polylines :-



6. Write out all the segments to a CSV file.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search