Get Weather Data Documentation

Release 0.1.29

Suriyan Laohaprapanon, Gaurav Sood

July 17, 2016

Contents:

Get Data from Weather Station Nearest to a Zip Code using the NOAA Web Service

Get data from nearest weather station given a list of zip codes and date (see sample input file for the format in which data are expected) using the NOAA webservice. The script appends all the weather data from NOAA along with the GHCND id, name, lat. and longitude of the weather station from which the data are being gotten to the input file (see sample output file).

1.1 Using NOAA-Web

- To get started, clone this subfolder from the repository:

```
git clone https://github.com/mfbx9da4/git-sub-dir.git
cd git-sub-dir
python get_git_sub_dir.py soodoku/get-weather-data/noaaweb
cd noaaweb
```

- The script needs an API token from NOAA. You can get a token from the NCDC site.

- Before running the file, open `noaaweb.py` in a text editor and replace `NCDC_TOKEN` with your NCDC token.

- The default output file name is `output.csv`. To specify a custom output file name, pass `-o outfilename_of_choice`.

- The script keeps track of the rows that have been processed. (It does so by taking row number from the output file as the start.) Thus, if halted in between, it will start with the last processed row.

Example

`python noaaweb.py samplein.csv -o sampleout.csv`

1.2 Note

Requests to NOAA API "often return nothing. It isn't clear why. The documentation doesn't say whether the search for the closest weather station is limited to X kilometers because without that, one should have data for all zip codes and all dates. Nor does the API bother to return how far the weather station is from which it got the data." (From Bad Weather: Getting weather data by zip and date). See for instance sample output file produced using the NOAA API.

1.3 License

The script is under the MIT License.

zip2ws: Find weather stations 'nearest' to zip codes

2.1 What it does:

1. Finds (certain kinds of) weather stations "nearest" (within a certain distance, or X number of closest) to each zip code centroid

2. Finds centroids of zip codes using Google API

2.2 Additional Details

Weather stations come in lots of varieties. We limit ourselves to weather stations of the following four kinds:

1. GHCND stations list. For current list, see: NCDC GHCND Stations List

2. ASOS stations list. For current list, see: NCDC ASOS Stations List

3. COOP stations list (Active only). For current list, see: NCDC COOP Stations List

4. USAF-WBAN stations list. For current list, see: NCDC ISH Stations List

2.3 Fields

- `zip, lat, long, city, state, zipcodetype, locationtype, location, decommisioned, taxreturns, estimatedpopulation, totalwages` from federalgovernmentzipcodes.us:

- gm_lat/gm_long: lat./long. of centroids of zip codes via Google API.

- diff: distance in meters between Google API estimated centroid of zip code and lat/long that comes with the database.

- list of stations: ordered from closest to furthest

- stX_id: station id

- stX_name: name of station

- stX_distance: distance to zip centroid

2.4 Running the script

To run the script, you will need to install two Python libraries:

- pygeocoder To install, you can simply use: `$ pip install pygeocoder`
- requests To install, you can simply user: `$ pip install requests`

Don't forget the inventories directory that contains the station files and zip csv that is imported. The inventories folder should be in the same folder as the script.

Start using the script by creating and importing the database. Do so by running:

Next task is to update the closest weather stations table. This you can do by executing...

This task uses the Google lat/long. If you want them to use other lat/long,

NOTE: If you interrupt the script inbetween and restart it again, the script will start processing from where it left off.

If you want to find a set number of closest stations, specify the type and number of weather stations. For instance, to find 5 GHCND statons, 3 COOP stations, and 2 USAF stations, run:

To find all weather stations within 30KM and organized by closest to furthest,

To export results to a CSV file, "closest.csv", run..

To find out centroids of zip codes using Google Maps Geocoding API, use

Keep in mind that Google Maps Geocoding API usage limit is 2,500 Query/Day/IP Address. So you can quickly run into the limit. The script will raise the exception "OVER_QUERY_LIMIT" if the limit is breached. But do not fear. You can run the script multiple times to code a greater number of zip codes. If you are unahppy with the results, use the option: –clear-glatlon to clear exist data.

zip2wd_mp: Get Weather Data For a List of Zip Codes For a Range of Dates (Multi-processing version)

Given a zip code and a date or a range of dates, it gets weather data (you get to specify which data) from the closest weather station from which the data are available. If given a range of dates, it fetches all the specified columns for each of the days in the intervening p period.

How it does it:

This script is based of the script that calculates nearest weather station based on variety of metrics.

You can use a variety of options to choose the kinds of weather stations from which you want data. For instance, you can get data only from USAF stations.

The script features on demand data downloads. So it pings the local directory and sees if weather data for a particular day and time are present and if they are not, then it tries to download it from the NOAA website. On occassion the script may run into bandwidth bottlenecks and you may want to run the script again to download all the data that is needed.

3.1 Prerequisites:

1. zip2ws.sqlite is based off finding the nearest weather station project.

2. Input File Types:

    (a) Basic: The input file format should be CSV and should contain 6

    columns with following columns names: |`uniqid, zip, year, month, day`| See sample-input-basic.csv for sample input file.

    (b) Extended: The input file format contain 9 columns with the

    following columns names: | 'uniqid, zip, from.year, from.month, from.day, to.year, to.month, to.day | See sample-input-extend.csv for sample input file.

3. Column Name File: This file contain list of weather data columns

    chosen for output file. | The column names begining with character '#' will not be appear in the output file. | (see column-names.txt for sample file)

    For what these column names stand for, see column-names-info.txt

4. GHCN Weather Data in SQLite3 database: These files create by a script import-db.sh for each year.

    e.g. for year 2000

```
    cd data
    ./import-db.sh 2000
```

The script will download daily weather data (GHCN-Daily) from NOAA server for year 2000 and import to SQLite3 database file (e.g. `ghcn_2000.sqlite3`)

3.2 Configuration file

There are script settings in the configuration. zip2wd.cfg

```
[manager]
ip = 127.0.0.1
port = 9999
authkey = 1234
batch_size = 10

[worker]
uses_sqlite = yes
processes = 4
nth = 0
distance = 30

[output]
columns = column-names.txt

[db]
zip2ws = zip2ws.sqlite
path = ./data/
```

- `ip` and `port` - IP address and port of manager process that the worker will be connect to.
- `authkey` - A shared password which is used to authenticate between manager and worker processes.
- `batch_size` - A number of zipcodes that manager process dispatch to worker process each time.
- `uses_sqlite` - Uses weather data from imported SQLite3 database if `yes` (recommend for speed) or download weather data for individual weather station on demand if `no`
- `processes` - A number of process will be forked on the worker process.
- `nth` - Search within n-th closest station [set to `0` for unlimited]
- `distance` - Search within distance (KM) [set to `0` for unlimited]
- `column` - A column file that contains list of weather data column to be output
- `zip2ws` - SQLite3 database of zip codes and weather stations
- `path` - Path relative to database files

3.3 Usage

3.3.1 Manager process

```
usage: manager.py [-h] [--config CONFIG] [-o OUTFILE] [-v] inputs [inputs ...]

Weather search by ZIP (Manager)

positional arguments:
  inputs                CSV input file(s) name

optional arguments:
  -h, --help            show this help message and exit
  --config CONFIG       Default configuration file (default: zip2wd.cfg)
  -o OUTFILE, --out OUTFILE
                        Search results in CSV (default: output.csv)
  -v, --verbose         Verbose message
```

3.3.2 Worker process

```
usage: worker.py [-h] [--config CONFIG] [-v]

Weather search by ZIP (Worker)

optional arguments:
  -h, --help       show this help message and exit
  --config CONFIG  Default configuration file (default: zip2wd.cfg)
  -v, --verbose    Verbose message
```

3.4 Example:

1. Run manager process search weather data for the input file sample-input-extend.csv

```
python manager.py sample-input-extend.csv
```

   The default output file is `output.csv`

2. Run worker process

```
python worker.py
```

   The manager will dispatch job (list of zip codes and date range) to the connected workers. The worker process also forks a number of process (specify by `processes` in the configuration file) to search the weather data for each zip code and put back the results to the manager process.

   We can have multiple workers run on same or difference machine.

3.5 Output

For each day you get weather columns that you mention in column-names
See column-names-info for details
SID = Station ID
type= Type of Station
Name = Name of Area
Lat = Latitude
Long = Longitude

Nth = N on the list of closest weather stations

Distance = Distance from zip code centroid to weather station lat/long

in meters

Indices and tables

- genindex
- modindex
- search