Exploring Data in R

Jason Thomas

November 15th, 2018

In collaboration with the WHO VA Reference Group

Quick Recap & Introduction

- Yesterday, we saw that. . .
  - Python is very powerful and flexible
  - there are packages, like `pandas`, that provide many useful tools
  - examples include system commands (e.g., creating directories and running other programs)
- Python's capabilities make it useful for processing verbal autopsy records.

Motivation

- Processing Verbal Autopsy Records
  1. Download records from ODK Aggregate with Briefcase
  2. Use `openVA` to assign causes of death
  3. Store the CoD results – e.g., in a DHIS2 or a database

  - still need data checks! (ideally this occurs before starting the Pipeline)
  - still need to check that the results make sense!

- This process can be automated using the `openVA Pipeline`
  - Python3.5 package – `openva-pipeline` – with tools for performing each step for processing VA data
    - includes a tool for running all of the steps
  - There is an SQLite database that holds configuration information and CoD results
  - The pipeline can be automated to run on a set schedule (e.g., once a week)

openVA Pipeline

- The openVA Pipline package is available at the *Py*thon *P*ackage *I*ndex: https://pypi.org/project/openva-pipeline/
- The documentation is located at *Read the Docs*: https://openva-pipeline.readthedocs.io/en/latest/
- Source code lives on Git Hub: https://github.com/verbal-autopsy-software/openva_pipeline

openVA Pipeline: installation

- Python packages can be installed with a program called `pip3`
- If dependencies are specified, `pip3` will install these packages as well
- The `openva-pipeline` packages depends on `pandas`, `pysqlcipher3`, and `requests` (these dependences also have dependencies as well)
- In a terminal, use the following commands to list the installed packages and to install the `openva-pipeline` package (and its dependencies if necessary)

```
pip3 list
pip3 install openVA-pipeline --user
```

  - the `--user` option installs the package in the user's library (not accessible by all users on the computer)

openVA Pipeline: functionality

- Transfer Database
  - Information needed for Pipline, ODK, openVA, DHIS2 (e.g., ODK aggregate username and password)
  - CoD results combined with VA data
  - Event Log (includes error messages)
- The Transfer Database can be created with the following Python3 commands.

```
import openva_pipeline as ovaPL
ovaPL.createTransferDB("Pipeline.db",
                       "/home/Jason/Pipeline/",
                       "key1234")
```

  - `Pipeline.db` is the name of the database
  - `/home/tot/Pipeline` is the directory where the database is created
  - `key1234` is the encryption key
- DB Browser is a useful GUI for interacting with the Transfer Database

openVA Pipeline: functionality (continued)

- The \texttt{openva_pipeline} can be run with the following command

```
import openva_pipeline as ovaPL
ovaPL.runPipeline("Pipeline.db",
                  "/home/Jason/Pipeline/",
                  "key1234")
```

  - `Pipeline.db` is the name of the database
  - `/home/tot/Pipeline` is the directory where the database is created
  - `key1234` is the encryption key

openVA Pipeline: modules

- TransferDB –
  - fetch information needed by ODK, openVA, and DHIS2
  - store CoD assignments and VA records in local database
- ODK – run ODK Briefcase and merge with previous export if necessary
- openVA – create and run an `R` script to `openVA`
  - includes capabilities for running SmartVA
- DHIS – (optional) post records to the *DHIS* VA Program

Pipeline: demonstration

For the demonstration will will use the ODK Aggregate and DHIS2 servers from SwissTPH

- ODK Aggregate
- DHIS2