QAMyData: an open source for tool for checking (and teaching) data quality

Louise Corti
Service Director Data Publishing and Access

IASSIST 2019
Sydney, Australia

UK Data Service

UK • DATA ARCHIVE

University of Essex

- 18 month project funded from Research Methods funding stream

- Develop a light weight, open-source tool for quality assessment of research data - a 'data health check' tool that automatically identifies the most common problems in research utilising quantitative methods

- Undertake evaluation of the tool and tests

- Help support integration of the tool into general data repository workflow, through liaison and testing

When does data need quality assessing?

Data publishing

- A researcher submitting data to a repository
- A repository for checking quality
- Peer review of published analysis for a journal

Using data

- A researcher using a new data source
- Preparing data for students learning about quality

When we want data to be healthy and safe

Data review

- Quality issues arise in the data description and data itself
- Datasets to be shared require a Privacy Impact Assessment (e.g. GDPR)
- Privacy level defines its legal gateway for access

✓ Share clean and well documented data

✓ Share data under the right conditions

UK Data Service Data Access Policy

Open	• Download/online access under open licence without registration
Safeguarded	• Download/online access to authorised and authenticated users
Controlled	• Remote or safe room access for registered users whose projects have been approved and who have received specialist training

The burden of checking data

- Get to know data:
  - incorrect, missing, inconsistent values
  - check structure
  - locate issues and decide how to treat them
- Many data creators/publishers use manual methods:
  - Statistical software commands
  - Mostly eyeballing
  - Rarely are variable-specific integrity checks used
- Treatment: clean data or flag errors

Checking data

- Are data values correct? Do they make sense?

pregnant	sex	age in years	occupation
y	female	29	2018-07-16: 12:37
n	female	15	2018-07-16: 12:38
y	female	32	2018-07-16: 12:39
n	male	37	2018-07-16: 12:40
n	female	-10	2018-07-16: 12:41
n	female	51	2018-07-16: 12:42
n	male	22	2018-07-16: 12:43
n	male	126	2018-07-16: 12:44
n	male	28	2018-07-16: 12:45
y	male	31	2018-07-16: 12:46
n	male	42	2018-07-16: 12:47
y	female	37	2018-07-16: 12:48

timestamp
2017-01-22: 10:15
201701241016
2017 01 22 10-17
2017-01-22-1018
2017 01 22 10 19

- Check frequency distributions for erroneous outliers, formats

Missing data…

age	income (£)	education_level	occupation type
40	52,000	postgraduate	professional
34	35,000	A-level	non-professional
25		GCSE	non-professional
30		GCSE	non-professional
28	22,000	GCSE	non-professional
55	37,000	graduate	professional
45		postgraduate	professional
33		A-level	non-professional
41	53,000	postgraduate	professional
43		postgraduate	professional
27	22,500	GCSE	non-professional

Data review: what to look for

- Basic file checks
- Metadata issues
- Data integrity Issues
- Disclosure control

Scoping the tests

- In-house procedures for data checking
- Prepared ‘dirty’ test datasets for evaluation
- Reached out to other archives/data publishers to gather information on their own QA checks
- Feedback from tool use in early training

Basic file checks

File opens

Bad filename check, regular expression pattern - RegEx

Metadata checks	
Missing variable labels	
Odd characters in variable names or labels, and value labels	Specified by user
Maximum length of variable labels	Var label > 79
'Maximum length of categorical value labels	Value label > 39
No label for user defined missing values	
Invalid variable names	

Data integrity checks	
Number of cases and variables	
Check for duplicate IDs	
Values outlying the listed code values	User specifies variables and expected range
Odd characters in string data	Specified by user
% of values missing ('Sys miss' and undefined missing)	25% of values missing
Subset by variables	User selects variables
Subset by row e.g. random slices for big or continuous data	User selects first N rows or random slice

Disclosure checks	
Unique values identifying disclosure risk (freq of categorical vars or numeric values)	Threshold value > 5
Direct identifiers using RegEx pattern search	Run separately for postcodes, telephone numbers etc.

Key-identifiers and non-identifying variables (combinations of key variables) – we use and train on sdcMicro

Formats and test selection

- Accepts: SPSS, Stata, SAS, CSV
- Configure data type, threshold, modular design
- `# Checks can be commented or omitted to exclude them from the checks to be run`

```
qamd run "I:\dcmagd\QAMD\data\teaching    data%set.sav" --include-locators --output-format html --output results_teaching_data.html --config "I:\dcmagd\QAMD\config\default.toml"
```

```
# Variable Configuration

[variable_config.odd_characters]
setting = ["!", "#", "  ", "@", "ë", "ç", "ô", "ü"]
desc = "Variable names and labels cannot contain certain 'odd' characters."

[variable_config.missing_variable_labels]
setting = true
desc = "Variables should have a label."

[variable_config.label_max_length]
setting = 79
desc = "Variable labels cannot exceed a max length”
```

RegEx

```
# Checking for specific patterns by using RegEx (as this step is resource intensive it has
been commented out from the initial config file, please see the User Guide for more
information about RegEx and how to run this step)
#[value_config.regex_patterns]
#setting = [
  # Simple Email address Regex
  # "^([\\w\\.\\-]+)@([\\w\\-]+)((\\.(\\w){2,4})+)$"
  # UK post code regex
  # "([Gg][Ii][Rr] 0[Aa]{2})|((([A-Za-z][0-9]{1,2})|(([A-Za-z][A-Ha-hJ-Yj-y][0-9]{1,2})|(([A-
Za-z][0-9][A-Za-z])|([A-Za-z][A-Ha-hJ-Yj-y][0-9]?[A-Za-z])))))\\s?[0-9][A-Za-z]{2})",
  # Email addresses as per RFC 2822
  # "((([a-zA-Z0-9!#$%&'*+/=?^_`{|}~-]+(\\.[a-zA-Z0-9!#$%&'*+/=?^_`{|}~-]+)*)|(\"(([\\x01-\
\x08\\x0B\\x0C\\x0E-\\x1F\\x7F]|[\\x21\\x23-\\x5B\\x5D-\\x7E])|(\\[\\x01-\\x09\\x0B\\x0C\
\x0E-\\x7F]))*\"))@(([a-zA-Z0-9!#$%&'*+/=?^_`{|}~-]+(\\.[a-zA-Z0-9!#$%&'*+/=?^_`{|}~-]+)*)|(\
\[(([\\x01-\\x08\\x0B\\x0C\\x0E-\\x1F\\x7F]|[\\x21-\\x5A\\x5E-\\x7E])|(\\[\\x01-\\x09\\x0B\
\x0C\\x0E-\\x7F]))*\\])))"
#]
#desc = "Values matching a regex pattern fail the check."
```

Reporting

QAMyData

mtcars.dta

Raw Case Count: 32
Total Variables: 12
Created At: 1519986540
Last modified at: 151
File Label:
File Format Version: 1
Compression type: No

Name
Variable odd charac
Date format
Value label max leng
Variable label max l
Missing variable lab
Variables with uniqu
Spellcheck
Value defined missi
System missing ove
Value odd character

Missing variable labels

# (limited to 1000)	Variable (Index)	Row Index
1	gear (10)	-
2	hp (4)	-
3	mpg (1)	-
4	am (9)	-
5	qsec (7)	-
6	model (0)	-
7	vs (8)	-
8	carb (11)	-
9	disp (3)	-
10	cyl (2)	-
11	drat (5)	-
12	wt (6)	-

Software decisions

Use existing open source libraries where possible

- SPSS Java libraries – IBM is licenced and difficult to deploy, Dext project
- Stata Java libraries – cross-version implementation not good
- R libraries – Haven based on Readstat – difficult to deploy, server licence, performance
- Python libraries – no good stable libraries for statistical packages
- Readstat

Using the ReadStat library

- A command-line tool and C library for reading files from popular stats packages
- Originally developed for Wizard for free stat data analysis on a Mac
- Supports SPSS, Stata and SAS files – new & old formats
- In active development since 2012, continually receiving security patches and bug fixes from the open source community
- Currently used by the R library Haven, part of the Tidyverse collection of R packages for data science

RUST programming language

- Tried the following wrappers: Java, Clojure, R, Python
- RUST from the Mozilla Foundation for improving the Firefox browser: an environment that demands things just 'work'
- Performance: Rust generates executables that run very fast, without needing to write low level C code; easily integrates with other languages
- Reliability: enables the developer to eliminate many classes of bugs at compile-time
- Productivity: has great documentation, a friendly compiler with useful error messages, and excellent tooling

Deployment

- Downloadable to run on Linux, Windows, Mac
- Simple to install and deploy from our Github
- Lightweight to run and set tests - edit config file
- Wiki with documentation
- Space for suggesting new tests
- Released under Creative Commons Attribution-NonCommercial 4.0 International License

Tool evaluation

First tier of evaluation

- UKDS data curation staff
- Peer data repositories in our international network

Second tier:

- University repositories - introductory webinar and visits
- Data owners,, researchers, data managers, quant. methods lecturers, journal publishers who run data peer review
- Voluntary testing - YOU!

Advocate data publishers to develop a Data Quality Profile with stated thresholds

Resources

- Table of Available Tests
- Download and Run Guide
- User Guide – step by step for editing config. file (set your own thresholds and searches)
- Teaching resources, slides, exercises and test data
- A blog on its way

https://www.ukdataservice.ac.uk/about-us/our-rd/qamydata.aspx

Acknowledgements

Thank you to my wonderful QAMyData team:

- Myles Offord: open source developer
- Jon Johnson: lead specs and development
- Cristina Magder: teaching materials and user guides
- Anca Vlad: input into tests and testing

And to ADA's Adam Zammit for adding an Open Source front end!

Contact

Louise Corti

corti@essex.ac.uk

@LouiseCorti

UK Data Service

UK • DATA ARCHIVE

University of Essex