Technical Notes

from Data Ingestion to Profiling and Preparation

(Data Quality in R)

Carlo Fanara

Summary

The data science workflow (from ingestion to presentation of the results) includes data preparation. Examples are constructed with a representative variety of the data - mostly publicly available - deploying some R language libraries taken from CRAN, and used in the context of a Laptop platform. Although it is impossible to generalize to every possible type of data, these steps, preceding the ML modelling, are condensed in a single script and aim at a uniform and repeatable analysis.

1. Validated data – upon successful profiling, in a form usable for further analysis;

2. The result

Contents

[1 Introduction: R for Data Science 3](#_Toc519261224)

[1.1 R libraries on CRAN and data used 3](#_Toc519261225)

[2 The DS workflow 4](#_Toc519261226)

[2.1 Data Science steps 4](#_Toc519261227)

[3 The workflow in a script / function 7](#_Toc519261228)

[3.1 Imports and other fast read outs 7](#_Toc519261229)

[3.2 Data validation 7](#_Toc519261230)

[3.3 Data Dictionaries 8](#_Toc519261231)

[3.4 Data Exploration (“EDA”) 8](#_Toc519261232)

[3.5 Missing values 8](#_Toc519261233)

[3.6 Outliers and Anomalies 8](#_Toc519261234)

[3.7 Unbalanced data 8](#_Toc519261235)

[3.8 Feature engineering 8](#_Toc519261236)

[3.9 Dimensionality reduction 9](#_Toc519261237)

[3.10 Standardization and normalisation 9](#_Toc519261238)

# Introduction: R for Data Science

The typical data science workflow (from data ingestion down to presentation of the results) is made by a series of steps which include the data inception and ingestion, and goes through various formal and semantic validations, until “profiled” data can be considered as consolidated and thus used for analytics and data science in a repeatable way.

We use a few sources of data, from older and ‘want to-be projects’ and in all cases, realistic – “market” data – are for the typical “laptop volumes” ~O(GB), e.g. sometimes “higher than laptop”, would require use of a “big” data platform infrastructure, not dealt with here.

R libraries are taken from CRAN with exception of a few – stated explicitly. The results of this work is summarized in the present text doc, in a summary .pptx file and in a series of functions callswithin a sequential script on **https://github.com/cfanara/IDA**

A similar work will be done for several other tools available in the open source arsenal, notably with Python (see analogous separate document).

## R libraries on CRAN and data used

As it is impossible and useless to list all existing packages, in the following, we use the latest libraries for each of the steps in the portion of the workflow as they appeared and were tested at the date of writing. The selection is indicated in Table I below. The data references and dictionaries are in the annex after the references.

We should incorporate some steps according to the requirements of the GDPR – specifically Art.32 – we may add a section on this [notably about ideas of implementing an GDPR KPI in the sort of a ‘semaphore’ or a red-yellow-green speedometer to allow or else -block – at least -halt – further analysis].

# The DS workflow

## Data Science steps

The entire DS workflow phases can be summarized as follows (numerous variations exist, also depending on the nature of the problem and further breakdowns are also possible for each step):

1. Data collection / Import (structured) text data (text encodings)

2. Cleaning of data (« tidiness »)

3. Visualization/Graphical displays of data

4. Classification or clustering? Categorical or quantitative? Mining

5. Choice of Modelling and evaluation metrics

6. Automatic classification using predictive modelling and or forecasting

7. Reporting

The ones in green are part of the present document. Visualization is partially covered to the extent helpful in revealing the main data properties ad potential issues. It may be considered part of the data Exploration (EDA, all of this is debatable and anyway, but the whole process is typically cyclic].

|  |  |
| --- | --- |
| *Data workflow steps considered in this note* | The group of operations meant to obtain “consistent data” is the topic of this document.  A more recent discussion on the subject may be found in Wikham[[1]](#footnote-1).  A variant for unstructured data (as opposed to structured data presented here) or else, for Time Series specifically, is described elsewhere.  Moreover, each of the steps can be broken down in sub-step like in Table I where we enlist the libraries and fill for description/notes along with the progress in the workflow.  Note that there might be some discrepancies with respect to the generally accepted sequence, but this is due to the specificity of some of these libraries (for example because encompassing functions which embrace more than one step at once). |

Referring to table I, the libraries in the column whose link points to the [CRAN](https://cran.r-project.org/) official R repository, are either tested (e.g. they passed some test – green) on the examples given, or suggested for further test. ‘Tests’ here means a generic “it’s working” for the data used.

Benchmarking, when necessary in specific cases – will also be indicated in color.

Failed is caused by an error condition (abort like in the first case, see table (and I still report it because the library is useful, but might require troubleshooting steps which are not yet undertaken and anyway not reported here).

Finally, a sequence of calls to these libraries is included in the script, posted on the GitHub.

**Table I – Data ingestion and profiling steps withs corresponding R libraries**

|  |  |  |  |
| --- | --- | --- | --- |
| **Activity** | **Description purpose** | **Library/ies (deployed on which data)** | **Notes** |
| 1. Data Acquisition / Ingestion | Readout of data from external (web) or other internal source (within company, PC, laptop, etc.) | [rio](https://cran.r-project.org/web/packages/rio/index.html) (*SFO\_tr*)  xlsx  xLconnect | Quite a few data read libraries, see CRAN’s view.  The general wrapper, rio, has a dozen or so methods for different file formats. The ‘fst’ is the quickest known to date but we encountered errors, we need to troubleshoot this. Some Excel are used especially when some structuring in different data architecture is wished (we don’t report all of these!)  This section could be superseded via a dedicated one on “ingestion”.  Xlsx / xLconnect might be useful for projects like ELIA, where all data seems provided to us in Excel form |
| 2 Data Definitions | Data dictionaries - result is a data glossary or dictionary covering data elements definitions | [dataMeta](https://cran.r-project.org/web/packages/dataMeta/index.html) *(SF\_tr)* | Note that general functions can provide similar info. The lib gives more but it is not evident how the third step works, especially if many possible values can be attributed to a category – check with other data…: |
| 3. Data Profiling | check for formal and semantics discrepancies, provide stats, both for future viualisations and analyses. | [Validate](https://cran.r-project.org/web/packages/Validate/index.html) ()  [Janitor](https://cran.r-project.org/web/packages/janitor/index.html) (SFtrain)  [DataMaid](https://cran.r-project.org/web/packages/dataMaid/index.html) ()  [DataCheck](https://cran.r-project.org/web/packages/datacheck/datacheck.pdf) ()  [DataExplorer](https://cran.r-project.org/web/packages/DataExplorer/DataExplorer.pdf) () |  |
| 4. Data Standardisation | Get standardisation and normalisation. Note that these may be part of later modelling. Standardised data to be stored separately from the original data. | [Standardize](https://cran.r-project.org/web/packages/standardize/index.html) () | Options in *caret* exist as well, and usually as the most important standardization is z-transformation, we don’t need special libraries  Can argue about standard use of ‘z’ (assumes gaussian) |
| 5. Data Integration | Data from various sources to be linked and correlated easily. Data prepared and stored to be used efficiently by analytics | NA | For now, integration exist in domain specific environments like the Bioconductor via specific libraries |
| 6. Data Cleaning and Matching | Further clean and match the data to enhance the ability to integrate or link the data. Cleaning and matching using automated procedures, manual intervention. | [dplyr](https://cran.r-project.org/web/packages/dplyr/index.html)  [tidyr](https://cran.r-project.org/web/packages/tidyr/index.html)  [reshape2](https://cran.r-project.org/web/packages/reshape2/index.html)  [lubridate](https://cran.r-project.org/web/packages/lubridate/index.html)  [timekit](https://cran.r-project.org/web/packages/timekit/index.html)  tsbox | These can be seen as data Wrangling libraries and are used to help aggregations and selections for EDA purposes. tidyr can be seen as alternative to reshape2. lubridate handles time and date formats, notoriously problematic.  For the timekit, this is one of the several time series library, mentioned here because it provides a time signature (as well as coercion from / to other time series formats). Similarly, tsbox promises to solve the perennial time format problem with TS in R |
| 3.5 missing values |  |  |  |
| 3.6 Outliers and anomalies |  |  |  |
| 3.7 unbalanced data |  |  |  |
| 3.8 Feature engineering |  |  |  |
|  |  |  |  |

*Color code: green, tested individually in this instance; purple: proposed in a single script, see section 3*

A further aim of this analysis is to ease the integration of these steps – grouping a selection of these by using the corresponding libraries and code chunks – so that in summary, the sequence completes at point 3, where the data are now profiled, formally correct, cleaned, consistent and for which basic statistics has been provided (e.g. general informative – non-domain specific). This is what we consider as data ready for the modelling and analysis steps.

It would be nice to incorporate other libraries which exist in the [Bioconductor](https://www.bioconductor.org/) standards, e.g. for data of “omic” origin (specifically genomics). These include [encapsulation of data](https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1455-1), path profiling and some niceties on the data structure visualization and gene [pathways](http://pathview.r-forge.r-project.org/). However, these are too specialized and / or corresponding but more general-purpose libraries have not been published yet.

Another section which might be useful is the one dealing with other formats, specifically JSON and, perhaps .avro, although I think these would be treated in the big data infrastructure rather than at the R level.

# The workflow in a script / function

Ideally, one would like to use a single overall function to prepare our data up to the point where a modeling step can be implemented. However, it is difficult to be that general and modifications might be necessary in order to prepare the data according to the varying input.

Still we attempt this, applying in sequence the selection of libraries indicated in the table above. If we can maintain modularity of the steps, we can hope to plug-in – or out – the steps as necessary. We use the script in the appendix – and the Github as a prototype to check the mentioned libraries in sequence. It is probably impossible to be that general, but modifications can be imagined on the implemented version to accommodate for the variability of the input data. Also, the order in the sequence may vary a bit.

## Imports and other fast read outs

For **the Ingestion**, there are a coupler of useful libraries. The first, **rio**, is a wrapper of a number of other reading methods, and frees the user from identifying the correct one (for example the various ASCII or excel or other text, sequential or in blocks, etc.) and deploys a generic import function.

A potentially very useful one would be **fst** which is the quickest way to read data. We encountered errors and as this need troubleshooting, we do not cover it here.

Libraries for the ingestion form multiple sheet Excel files are (among many others) xLconnect and Xlsread.

## Data validation

**For the validation** of data, **validate()** offers two interesting the possibilities:

1. To construct a validator where rules can be inputted so that the dataset at hand can be compared to it. The rules are defined and saved into a “validator” object
2. To compare a different version of a dataset one another, one version could be taken as a reference dataset. In practice, this could be a periodic verification performed upon subsequent data ingestions

From the vignette of the library and the library documentation, we see that the general workflow in validate follows the following pattern:

• Define a set of rules using validator. This is done in a text file or in a .yaml file which gives more flexibility

• confront data with the validation rules. We tested the simple case of no additional reference data given (see function confront in the vignette for additional use of keys in the data)

• Examine the results either graphically or by summary

This has been done for the SFO data using the same rules deployed individually in the example file. [The references and pointers to the files (R scripts) are in the annex]

Still for validation, the **DataMaid package** is described as a Suite of Checks for Identification of Potential Errors in a Data Frame. This is also part of the data Cleaning Process. It can produce an overall doc about the data checked with the indication about all the tests performed and their results.

The need for a dictionary arises in the **datacheck** package where we use the dictionary as a checking tool for the data correctness. In fact, in the basic example provided with the vignette, two questions are asked for each of the variables and set within the dictionary object used to validate data: one is about their type, another is about their limiting values.

## Data Dictionaries

**For the Dictionary**, we may use **Datameta,** which however, seemsproblematic: individual values are shown as categories and listed, so first, need to be sure about the formats, which defies a bit the purpose of all this (we need a dictionary beforehand to define one?). Operationally, once a description I s entered, the user is prompted about a description (further one) about the one just entered. Unclear how/why this nested mechanism would be useful.

## Data Exploration (“EDA”)

## Missing values

A few libraries allow for the determination, visualization and imputation of missing values. Some of these do all three the things. We select here the VIM among these, because it also yields a visualization of the ‘holes’ (“NA’s”)

## Outliers and Anomalies

Here we consider data which break the ‘normality’ (outliers) without specifically considering the issue of searching for anomalies. The topic is described in a different doc and implemented elsewhere, but that workflow can be plugged in into the present workflow.

## Unbalanced data

Assuming we are not specifically looking for anomalies – but see above – or rare events (separate document) here we deal with the situation where in a classification – binary or multivariate – an unbalance exists between one majority class and one or several minority classes. It is arbitrary to decide where the unbalance sits in terms of values, but ratios between the minority class and the total of the observations of 10% or less are certainly ‘unbalanced’ . We do not discuss the theory as several sources exist [ ].

Methods are undersampling – penalizing the majority class/es, up sampling / promoting the minority class/es or combinations of the two. In addition, more sophisticated method exists (like ‘SMOTE’).

In R we have

## Feature engineering

By far the most utilized method is Principal Component Analysis (PCA) although several other methods – mostly linear exist (Factor analysis is one). An interesting possibility is to use quick and dirty ML implementation of Random Forest – say – in order to look for variable importance non linearly. Just use the score of a RF to determine which variables are less important.

## Dimensionality reduction

## Standardization and normalisation

1. Hadley Wikham, Tidy Data, Journal of statistical software, Vol.59 (2014), issue 10, DOI [10.18637/jss.v059.i10](http://dx.doi.org/10.18637/jss.v059.i10) [↑](#footnote-ref-1)