Generating a codebook for data tables

Daniel Manrique-Castano, Ph.D
Digital Research Alliance of Canada
daniel.manrique-castano@alliancecan.ca

Tuesday, August 20, 2024

Table of contents

Codebooks, also called dictionaries, are files that provide information about the contents and structure of data files or tables. In other words, they allow users to understand what information is contained in a data file. Since a well-formed data table is limited in the amount of information it can contain, the source of the data and the definitions of the variables must be in a separate document. Here we use the `codebookr` package [1] to generate codebooks and dictionaries in .docx format.

1 Install and load packages

We install the required packages:

```r
#install.packages("codebookr")
#install.packages("dplyr")
```

And load them into the current R project/session:

```r
library(codebookr)
library(dplyr, warn.conflicts = FALSE)
```

2 Load and explore the dataset

We use the `read.csv` function to load out example data table. This table correspond to research research data in experimental neuroscience borrowed from XXX

```r
data <- read.csv("data/Example_Data1.csv")
```

Then, we use the `glimpse` function from the `dplyr` package [2] to get a summary description of our table:

We identify 5 columns and 262 rows of data. The first four columns appear to be character values, and only the last column is numeric. Given the structure and labeling of the table, we cannot tell what its content or source is, which severely limits its reuse.

3 Generating a codebook file

To describe the table and label the variables, we use the `codebook' function from the`codebookr' package. First, let's see what we get when we run the function on the unlabeled table. We run the function and print the results to a .docx file

```
data_unlabeled_codebook <- codebook(data)
print(data_unlabeled_codebook, "data_unlabeled_codebook.docx")
```

The resulting file (see your project folder) shows basic (appropriate) basic statistics for each variable column. However, there is still no information about the content. In the following, we will use the `cb_add_col_attributes` function to provide the relevant information for the codebook.

First, to take full advantage of the function, we explicitly specify the type of variable for each column. This is where the author uses his knowledge of the data table (otherwise uknkown by the community):

```
data$MouseID <- as.character(data$MouseID)
data$DPI <- factor(data$DPI, levels = c("0D", "3D", "7D", "14D", "30D"))
data$Region <- factor(data$Region, levels = c("Perilesion", "Striatum",
"Cortex"))
data$Gfap <- factor(data$Gfap, levels = c("Low", "High"))
data$Pdgfrb <- as.numeric(data$Pdgfrb)

glimpse(data)
```

```
Rows: 262
Columns: 5
$ MouseID <chr> "Td12", "Td12", "Td12", "Td14", "Td14", "Td14", "Td16", "Td16"…
$ DPI     <fct> 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 14D, 14D, 14D,…
$ Region  <fct> Cortex, Perilesion, Striatum, Cortex, Perilesion, Striatum, Co…
$ Gfap    <fct> Low, Low, Low, Low, Low, Low, Low, Low, Low, Low, Low, Low, Lo…
$ Pdgfrb  <dbl> 316, 156, 35, 155, 91, 114, 410, 69, 74, 311, 72, 52, 604, 68,…
```

We see that the variable types have changed. Now we pass the `cb_add_col_attributes` function to each of the variables to provide informative descriptions that would otherwise be unknown to the community.

```
data__labeled_codebook <- data %>%
  cb_add_col_attributes(
    MouseID,
    description = "Unique animal identification number",
    source = "Experimental metadata",
    col_type = "categorical"
  ) %>%

  cb_add_col_attributes(
    DPI,
     description = "Days post-ischemia (time point in which the brain was
harvested)",
    source = "Experimental metadata",
    col_type = "categorical"
  ) %>%

  cb_add_col_attributes(
    Region,
    description = "Brain region imaged in the microscope. Cortex = ischemic
dorsolateral cortex, Striatum = ischemic ipsilateral striatum, Perilesion =
Healthy perilesion in the dorsolateral cortex",
    source = "Experimental metadata",
    col_type = "categorical"
  ) %>%

  cb_add_col_attributes(
    Gfap,
    description = "Gfap density measure by point pattern analysis. Possible
values = Low and High",
    source = "PPA analysis - Link_To_Analysis_Notebook",
    col_type = "categorical"
  ) %>%

  cb_add_col_attributes(
    Pdgfrb,
    description = "Number of PDGFRB+ cells estimated by quadrant counts",
    source = "PPA analysis - Link_To_Analysis_Notebook",
    col_type = "numeric"
  )
```

```
The following attribute(s) are being added to a variable in the data frame
for the first time: description, source, col_type. If you believe this/these
attribute(s) were previously added, then check for a typo in the attribute name.
If you are adding this/these attribute(s) for the first time, you can probably
safely ignore this message.
```

```
glimpse(data)
```

```
Rows: 262
Columns: 5
$ MouseID <chr> "Td12", "Td12", "Td12", "Td14", "Td14", "Td14", "Td16", "Td16"…
$ DPI     <fct> 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 7D, 14D, 14D, 14D,…
$ Region  <fct> Cortex, Perilesion, Striatum, Cortex, Perilesion, Striatum, Co…
$ Gfap    <fct> Low, Low, Low, Low, Low, Low, Low, Low, Low, Low, Low, Low, Lo…
$ Pdgfrb  <dbl> 316, 156, 35, 155, 91, 114, 410, 69, 74, 311, 72, 52, 604, 68,…
```

At this point, `glimpse' shows that the variable types have changed as intended. Finally, we pass the`codebook' function with additional arguments to describe the data table.

```
data_codebooktext <- codebook(
  df = data,
  title = "Example_Data1.csv",
  description = "This is the codebook for the Example_Data1.csv data table. This
table is derived from a neuroscience experiment in which mice were subjected
to cerebral ischemia and euthanized at different time points (DPI) to study
brain recovery. These data correspond to brain sections stained with antibodies
against PDGFR-B and GFAP. Using point pattern analysis (link to notebook), we
quantified the number of PDGFR-B+ cells in regions of low and high GFAP density."
)
print(data_codebooktext, "data_labeled_codebook.docx")
```

In this case, the codebook will be exported to the project folder. It is highly recommended that the user set specific/descriptive file names and save in the appropriate folder according to the project structure.

For alternative ways of labeling data using the `sjlaballed` [3] and the `labelled` [4] packages please check this [post](https://www.pipinghotdata.com/posts/2022-09-13-the-case-for-variable-labels-in-r/) and [presentation slides](https://github.com/shannonpileggi/context-is-king) by Shannon Pileggi.

4 References

```
sessionInfo()
```

```
R version 4.4.1 (2024-06-14 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 22631)

Matrix products: default
```

```
locale:
[1] LC_COLLATE=English_Canada.utf8  LC_CTYPE=English_Canada.utf8
[3] LC_MONETARY=English_Canada.utf8 LC_NUMERIC=C
[5] LC_TIME=English_Canada.utf8

time zone: America/Toronto
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] dplyr_1.1.4     codebookr_0.1.8

loaded via a namespace (and not attached):
 [1] utf8_1.2.4              generics_0.1.3          tidyr_1.3.1
 [4] fontLiberation_0.1.0    xml2_1.3.6              stringi_1.8.4
 [7] httpcode_0.3.0          hms_1.1.3               digest_0.6.36
[10] magrittr_2.0.3          grid_4.4.1              evaluate_0.24.0
[13] flextable_0.9.6         fastmap_1.2.0           jsonlite_1.8.8
[16] zip_2.3.1               crul_1.5.0              promises_1.3.0
[19] purrr_1.0.2             fansi_1.0.6             fontBitstreamVera_0.1.1
[22] textshaping_0.4.0       shiny_1.8.1.1           cli_3.6.3
[25] crayon_1.5.3            rlang_1.1.4             fontquiver_0.2.1
[28] withr_3.0.0             gfonts_0.2.0            yaml_2.3.9
[31] gdtools_0.3.7           tools_4.4.1             officer_0.6.6
[34] uuid_1.2-0              httpuv_1.6.15           forcats_1.0.0
[37] curl_5.2.1              mime_0.12               vctrs_0.6.5
[40] R6_2.5.1                lifecycle_1.0.4         stringr_1.5.1
[43] ragg_1.3.2              pkgconfig_2.0.3         later_1.3.2
[46] pillar_1.9.0            glue_1.7.0              data.table_1.15.4
[49] Rcpp_1.0.12             systemfonts_1.1.0       haven_2.5.4
[52] xfun_0.45               tibble_3.2.1            tidyselect_1.2.1
[55] rstudioapi_0.16.0       knitr_1.48              xtable_1.8-4
[58] htmltools_0.5.8.1       rmarkdown_2.27          compiler_4.4.1
[61] askpass_1.2.0           openssl_2.2.0
```

Bibliography

[1] B. Cannell, “codebookr: Create Codebooks from Data Frames,” 2024, [Online]. Available: https://cran.r-project.org/package=codebookr

[2] H. Wickham, R. François, L. Henry, K. Müller, and D. Vaughan, “dplyr: A Grammar of Data Manipulation,” 2023, [Online]. Available: https://cran.r-project.org/package=dplyr

[3] J. Larmarange, “labelled: Manipulating Labelled Data,” 2024, [Online]. Available: https://cran.r-project.org/package=labelled

[4] D. Lüdecke, “sjlabelled: Labelled Data Utility Functions (Version 1.2.0),” 2022, doi: 10.5281/zenodo.1249215.