Package 'DedooseR'

January 9, 2026

Title Monitoring and Analyzing Dedoose Qualitative Data Exports

Version 2.0.0.2

Description
Streamlines analysis of qualitative data exported from 'Dedoose' <https://www.dedoose.com>. Supports monitoring thematic saturation, calculating code frequencies, organizing excerpts, generating dynamic codebooks, and producing code network maps within 'R'.

License Apache License (>= 2)

URL https://abiraahmi.github.io/DedooseR/
https://github.com/abiraahmi/DedooseR

BugReports https://github.com/abiraahmi/DedooseR/issues

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, tidyr, ggplot2, knitr, DT, tibble, labelled, kableExtra, ggraph, igraph, wordcloud2, tidytext, purrr, haven, openxlsx

VignetteBuilder knitr

Config/testthat/edition 3

Suggests rmarkdown, testthat (>= 3.0.0)

NeedsCompilation no

Author Abiraahmi Shankar [aut, cre] (ORCID:
<https://orcid.org/0000-0001-8354-5353>),
Catalina Cañizares [aut] (ORCID:
<https://orcid.org/0000-0002-6854-5205>),
Francisco Cardozo [aut] (ORCID:
<https://orcid.org/0000-0002-1925-4954>),
Sabrina Stockmans [aut],
Pamela Morris-Perez [aut] (ORCID:
<https://orcid.org/0000-0003-3375-3458>)

Maintainer Abiraahmi Shankar <abiraahmi.shankar@nyu.edu>

Repository CRAN

Date/Publication 2026-01-09 00:20:07 UTC

Contents

`clean_data` *Clean and prepare qualitative excerpts for export*

Description

This function standardizes and cleans a dataset of qualitative excerpts coded by multiple coders. It standardizes column names, filters excerpts by a preferred coder hierarchy, converts code columns to logical (TRUE/FALSE), assigns descriptive variable labels, and optionally exports the cleaned data to Excel (`.xlsx`) or Stata (`.dta`) format. The function also returns a codebook containing variable names, labels, and data types.

Usage

```
clean_data(
  excerpts,
  preferred_coders,
  rename_vars = NULL,
  relabel_vars = NULL,
  output_path = NULL,
  output_type = c("none", "xlsx", "dta")
)
```

Arguments

`excerpts` A data frame containing excerpt-level data exported from Dedoose or a similar coding platform.

`preferred_coders` A character vector of coder names in order of preference. The function keeps the highest-preference coder for each unique `media_title`.

`rename_vars` An optional named list or `dplyr::rename()`-style mapping of variables to rename. For example, `list(new_name = "old_name")`.

`relabel_vars` An optional named list of new variable labels. For example, `list(old_name = "New label for var1", var2 = "Updated label for var2")`.

`output_path` Optional file path to save the cleaned dataset. If `NULL`, the data will not be saved to disk.

`output_type` A string specifying the export format. Must be one of:

- `"none"` – no file is written (default)
- `"xlsx"` – save as Excel file via `openxlsx::write.xlsx()`
- `"dta"` – save as Stata file via `haven::write_dta()`

Details

The function performs the following steps:

1. Standardizes variable names (lowercase, underscores instead of spaces).
2. Renames `excerpt_copy` to `excerpt` if present.
3. Removes columns ending with `"range"` or `"weight"`.
4. Detects code columns matching the pattern `"^code.*applied$"` and converts them to logicals.
5. Renames code columns with a `c_` prefix and assigns human-readable variable labels.
6. Filters to the preferred coder per `media_title`.
7. Applies default labels to key metadata variables (e.g., `excerpt_creator`, `media_title`).
8. Optionally renames or relabels variables via user-supplied arguments.
9. Drops columns that are entirely `NA`.
10. Generates a codebook summarizing variables, labels, and types.

When exporting to `.dta`, logicals remain stored as `TRUE`/`FALSE` rather than being coerced to 0/1. Variable labels are preserved in Stata format using the `labelled` and `haven` packages.

Value

A list with two elements:

`data` A cleaned data frame with standardized names, filtered coders, and labelled variables.

`codebook` A data frame with columns: `variable`, `label`, and `type`.

Examples

```
## Not run:
result <- clean_data(
  excerpts = excerpts_raw,
  preferred_coders = c("CoderA", "CoderB"),
  rename_vars = list(new_name = "old_name"),
  relabel_vars = list(old_name = "new variable label"),
  output_path = "cleaned_excerpts.dta",
  output_type = "dta"
)

# Access cleaned data and codebook
head(result$data)
head(result$codebook)

## End(Not run)
```

compare_saturation *Compare Code Saturation Across Threshold Sets*

Description

This function compares code saturation results from a code summary table (typically generated by `create_code_summary()`) against one or more threshold sets that define what constitutes "saturation" based on the number of excerpts (`count`) and proportion of media titles (`prop_media_titles`) where each code appears.

Optionally, the function can produce a faceted bar plot showing which codes meet each set of thresholds, with metrics plotted as counts, proportions, or both.

Usage

```
compare_saturation(
  code_summary,
  thresholds_list,
  output_type = c("tibble", "kable"),
  plot = FALSE,
  plot_metric = c("count", "prop", "both")
)
```

Arguments

`code_summary` A data frame or tibble, typically produced by `create_code_summary()`, containing columns: `code`, `count`, `n_media_titles`, and `prop_media_titles`.

`thresholds_list`
A named list of threshold sets. Each element should be a list with numeric elements `code_count` and `prop_media_title`. For example: `list(Liberal = list(code_count = 3, prop_media_title = 0.2), Strict = list(code_count = 5, prop_media_title = 0.5))`.

`output_type` Character string specifying the output type: either `"tibble"` (default) or `"kable"` for a formatted table.

`plot` Logical; if `TRUE`, generates a ggplot visualization of which codes meet each threshold set.

`plot_metric` Character string specifying which metric to plot:

- `"count"` — show excerpt frequencies only.
- `"prop"` — show proportions of media titles only.
- `"both"` — show both metrics with dual y-axes (counts on the bottom axis, proportions on the top axis).

Details

Each threshold set is applied independently. A code is considered to meet a given threshold set if both its excerpt count and proportion of media titles are greater than or equal to the respective thresholds.

Value

If `plot = FALSE`, returns either a tibble or kable table of results. If `plot = TRUE`, returns a list with:

- `results`: the tibble or kable table with logical columns indicating which codes meet each threshold set.
- `plot`: a ggplot2 object visualizing saturation across threshold sets.

Examples

```
# Example data: excerpts with coded logical columns
set.seed(123)
excerpts <- data.frame(
  media_title = rep(paste0("Interview_", 1:5), each = 3),
  code_A = sample(c(TRUE, FALSE), 15, replace = TRUE),
  code_B = sample(c(TRUE, FALSE), 15, replace = TRUE),
  code_C = sample(c(TRUE, FALSE), 15, replace = TRUE)
)

# Create a code summary table (from your package function)
code_summary <- create_code_summary(excerpts, output_type = "tibble")

# Define two saturation threshold sets
thresholds_list <- list(
  Liberal = list(code_count = 3, prop_media_title = 0.2),
  Strict  = list(code_count = 5, prop_media_title = 0.5)
)

# Compare saturation (table only)
compare_saturation(code_summary, thresholds_list)

# Compare and plot using proportions
res <- compare_saturation(code_summary, thresholds_list, plot = TRUE, plot_metric = "prop")
res$plot

# Compare and plot both metrics with dual y-axes
res2 <- compare_saturation(code_summary, thresholds_list, plot = TRUE, plot_metric = "both")
res2$plot
```

---

cooccur *Create a Code Co-occurrence Matrix and Network Plot*

Description

Builds a co-occurrence matrix showing how often qualitative codes appear together within the same unit (e.g., transcript, document, or media title). The function expects a coded dataset (excerpts) and returns both a formatted matrix and (optionally) a network visualization. The returned matrix can be displayed as raw counts or column-wise proportions, whereas the network plot always reflects the underlying raw counts.

Usage

```
cooccur(
  excerpts = NULL,
  min_bold = 10,
  scale = c("count", "prop"),
  output = c("kable", "tibble", "data.frame"),
  plot = TRUE,
  edge_min = 10,
  layout = "circle",
  edge_color_low = "lightgray",
  edge_color_high = "purple",
  node_color = "lightblue",
  use_labels = FALSE,
  codebook = NULL
)
```

Arguments

excerpts — Data frame containing coded excerpts, with a column named media_title and code columns prefixed with "c_".

min_bold — Minimum value for bold highlighting in HTML table output (if output = "kable"). Default is 10.

scale — Whether to display raw counts ("count") or column-wise conditional proportions ("prop") in the returned matrix. The network plot always uses raw counts. Default is "count".

output — The format of the co-occurrence matrix output. One of "kable", "tibble", or "data.frame". Default is "kable".

plot — Logical; whether to produce a network visualization. Default is TRUE.

edge_min — Minimum edge weight (in counts) for displaying connections in the plot. Default is 10.

layout — Graph layout for network visualization (passed to ggraph::ggraph). Common options include "circle", "fr", or "kk". Default is "circle".

edge_color_low, edge_color_high — Color gradient for edge weights in the plot. Default is "lightgray" to "purple".

node_color — Color for node points in the network plot. Default is "lightblue".

use_labels — Logical; if TRUE, replaces code variable names with descriptive labels from a provided codebook. Default is FALSE.

codebook — Optional data frame with columns:

- `variable`: the code variable name (e.g., `"c_family"`)
- `label`: the descriptive name for the code (e.g., `"Family Connectedness"`). Required when `use_labels = TRUE`.

Details

The function identifies columns beginning with `"c_"` as code variables. It computes co-occurrences by summing pairwise intersections of codes across all unique `media_title` units. The diagonal represents the marginal frequencies (the number of transcripts where each code appears).

The resulting matrix can be output as a tibble, a simple data frame, or a formatted HTML table via `knitr::kable`. If `plot = TRUE`, the function also returns a network visualization of code co-occurrences using `ggraph` and `igraph`. Edges are filtered via the `edge_min` threshold, and nodes without any remaining connections are removed from the plot.

Value

A named list with two elements:

matrix A tibble, data frame, or formatted HTML table of the co-occurrence matrix.

plot A `ggplot` object visualizing the co-occurrence network (if `plot = TRUE`).

Examples

```
# Example 1: Basic co-occurrence matrix from excerpts
df <- data.frame(
  media_title = c("Doc1", "Doc2", "Doc3"),
  c_hope = c(1, 0, 1),
  c_family = c(1, 1, 0),
  c_school = c(0, 1, 1)
)

result <- cooccur(
  excerpts = df,
  scale = "count",
  output = "tibble",
  plot = TRUE
)

result$matrix  # Co-occurrence matrix
result$plot    # Network plot

# Example 2: Use descriptive labels from a codebook and proportions in the table
codebook <- data.frame(
  variable = c("c_hope", "c_family", "c_school"),
  label = c("Hope & Optimism", "Family Connectedness", "School Belonging")
)

labeled_result <- cooccur(
  excerpts = df,
  use_labels = TRUE,
  codebook = codebook,
  scale = "prop",
  output = "kable",
  plot = TRUE
)

labeled_result$matrix
labeled_result$plot
```

create_code_summary *Create a Summary Table and Plot of Code Frequencies*

Description

Summarizes how often each qualitative code (represented by logical 0/1 variables) appears across excerpts or media titles. Optionally produces a frequency table and visualization of code distributions.

This function automatically handles Stata-labelled (haven_labelled) or numeric 0/1 variables by converting them to logicals. You can also pass in a custom codebook to apply human-readable code labels.

Usage

```
create_code_summary(
  excerpts,
  table_min_count = 1,
  table_min_prop = NULL,
  plot = FALSE,
  plot_min_count = NULL,
  plot_min_prop = NULL,
  output_type = c("tibble", "kable", "datatable"),
  exclude = NULL,
  plot_metric = c("count", "prop", "both"),
  fill_color = "steelblue",
  use_labels = FALSE,
  codebook = NULL
)
```

Arguments

excerpts
: A data frame containing at least one logical or 0/1 variable representing a code, and a column named media_title that identifies the source document or excerpt.

table_min_count
: Minimum number of excerpts required for a code to appear in the summary table. Default is 1.

`table_min_prop` Optional proportion threshold (relative to the maximum count) for including codes in the table. Default is `NULL`.

`plot` Logical; whether to generate a plot visualizing code frequencies. Default is `FALSE`.

`plot_min_count` Minimum number of excerpts required for a code to appear in the plot. Defaults to `table_min_count`.

`plot_min_prop` Optional proportion threshold (relative to the maximum count) for including codes in the plot. Defaults to `table_min_prop`.

`output_type` The format for the output table. One of `"tibble"`, `"kable"`, or `"datatable"`. Default is `"tibble"`.

`exclude` Optional character vector of variable names to exclude from analysis.

`plot_metric` The metric to visualize. One of `"count"`, `"prop"`, or `"both"`. Default is `"count"`.

`fill_color` Color for plot bars. Default is `"steelblue"`.

`use_labels` Logical; if `TRUE`, uses a supplied `codebook` to display descriptive labels for codes instead of variable names. Default is `FALSE`.

`codebook` Optional data frame with two columns:

- `variable`: the variable names in the dataset
- `label`: the corresponding human-readable label for each code. Required when `use_labels = TRUE`.

Details

The function first identifies all logical (or 0/1 numeric) columns in `excerpts` and calculates:

- `count`: total number of excerpts where the code is applied
- `n_media_titles`: number of distinct media titles containing the code
- `prop_media_titles`: proportion of media titles containing the code (relative to max)

The table can be output as a tibble, formatted table (`knitr::kable`), or interactive data table (`DT::datatable`).

When `plot = TRUE`, the function generates a ggplot2 bar chart showing either code counts, proportions, or both (dual-axis view).

Value

If `plot = FALSE`, returns a table in the selected `output_type` format. If `plot = TRUE`, invisibly returns a list with two elements:

table A table of summarized code frequencies.

plot A `ggplot` object visualizing the results.

Examples

```
# Example 1: Basic usage without a codebook
df <- data.frame(
  media_title = c("Doc1", "Doc2", "Doc3", "Doc4"),
  code_a = c(TRUE, FALSE, TRUE, TRUE),
  code_b = c(FALSE, TRUE, TRUE, FALSE)
)

create_code_summary(df, plot = TRUE)

# Example 2: Using a codebook for readable labels
codebook <- data.frame(
  variable = c("code_a", "code_b"),
  label = c("Community Engagement", "Policy Support")
)

create_code_summary(
  df,
  use_labels = TRUE,
  codebook = codebook,
  plot = TRUE,
  plot_metric = "both"
)

# Example 3: Excluding a code and outputting as datatable
create_code_summary(
  df,
  exclude = "code_b",
  output_type = "datatable"
)
```

recode_themes *Recode logical code variables and optionally relabel them*

Description

recode_themes() combines multiple logical (TRUE/FALSE) code variables into new composite variables. For each new variable, the function computes a logical OR across the specified source variables—meaning the new variable is TRUE when any source variable is TRUE. Optionally, descriptive labels can be supplied for the newly created variables, and a codebook summarizing the resulting dataset is returned.

Usage

```
recode_themes(data, recodes, relabel_vars = NULL)
```

Arguments

`data` A data frame, tibble, or haven-labelled data frame (for example, the output from clean_data() or a dataset read from a .dta file) containing logical code variables.

`recodes` A named list where each name is a new variable to create and each value is a character vector of existing variable names to combine. For example: list(c_help = c("c_support", "c_assist"), c_stress = c("c_anxiety", "c_pressure"))

`relabel_vars` Optional named list of variable labels for the new composite variables in the format list(new_var = "New variable label"). If omitted, the new variable names are used as default labels.

Details

The function first verifies that the specified source variables exist in the dataset. It then creates the new logical variables defined by recodes, assigns user-specified or default labels, removes the original source variables (unless one overlaps with a new variable name), and builds a codebook summarizing the recoded dataset.

Value

A list with four elements:

data_recode A data frame containing the updated dataset with recoded logical code variables.

codebook_recode A data frame summarizing variable names, labels (if available), and data types.

data_merged Alias for `data_recode` retained for backward compatibility.

codebook_merged Alias for `codebook_recode` retained for backward compatibility.

Examples

```
# Example dataset
df <- data.frame(
c_support = c(TRUE, FALSE, TRUE),
c_assist = c(FALSE, TRUE, TRUE),
c_anxiety = c(TRUE, FALSE, FALSE),
c_pressure = c(FALSE, TRUE, FALSE)
)

# Define recodes
recode_plan <- list(
c_help = c("c_support", "c_assist"),
c_stress = c("c_anxiety", "c_pressure")
)

# Run recode_themes() with new labels
result <- recode_themes(
data = df,
recodes = recode_plan,
relabel_vars = list(
c_help = "Mentions of helping or supporting others",
c_stress = "Mentions of stress or pressure"
)
)

# Extract recoded data and codebook
data_recode <- result$data_recode
codebook_recode <- result$codebook_recode

# View recoded dataset
head(data_recode)

# View codebook
head(codebook_recode)
```

---

set_saturation *Compute and Visualize Code Saturation*

Description

This function summarizes code counts and their proportional representation across media titles (e.g., interviews, focus groups, or other qualitative data sources). It can optionally produce a formatted table and/or a ggplot visualization showing saturation by code frequency or proportion.

Usage

```
set_saturation(
  code_counts,
  total_media_titles = NULL,
  table_min_count = 1,
  table_min_prop = NULL,
  output_type = c("tibble", "kable"),
  plot = FALSE,
  plot_min_count = NULL,
  plot_min_prop = NULL,
  plot_metric = c("prop", "count", "both"),
  fill_color = "steelblue"
)
```

Arguments

`code_counts` A tibble or data frame containing columns:

- `code`: the code label
- `count`: total number of excerpts coded with that code
- `n_media_titles`: number of distinct media titles (e.g., transcripts) in which the code appears. This object is typically generated by `create_code_summary()`.

`total_media_titles`
Optional numeric value indicating the total number of media titles. If `NULL` (default), the function uses the maximum value of `n_media_titles`.

`table_min_count`
: Minimum count threshold for including a code in the output table. Defaults to 1.

`table_min_prop`
: Minimum proportion threshold (relative to total media titles) for including a code in the output table. Defaults to `NULL` (no proportion filter).

`output_type`
: Character string indicating the output format for the table: either `"tibble"` (default) or `"kable"`.

`plot`
: Logical; if `TRUE`, produces a ggplot visualization. Defaults to `FALSE`.

`plot_min_count`
: Minimum count threshold for codes to include in the plot. Defaults to `table_min_count` if `NULL`.

`plot_min_prop`
: Minimum proportion threshold for codes to include in the plot. Defaults to `table_min_prop` if `NULL`.

`plot_metric`
: Character string indicating what to plot: `"prop"` for proportions, `"count"` for counts, or `"both"` for dual-axis plot. Defaults to `"prop"`.

`fill_color`
: Character string specifying the fill color for bars in the plot. Defaults to `"steelblue"`.

Value

- If `plot = FALSE`: returns a tibble (or kable table) summarizing code frequencies and proportions.
- If `plot = TRUE`: returns a list with two elements:
  - `table`: the filtered tibble
  - `plot`: a ggplot2 object.

Examples

```
# Example dataset
code_counts <- tibble::tibble(
  code = c("Belonging", "Resilience", "Stress", "Hope"),
  count = c(15, 10, 8, 5),
  n_media_titles = c(8, 6, 5, 3)
)

# Basic usage (returns a tibble)
set_saturation(code_counts)

# Apply count and proportion filters, return a kable table
set_saturation(
  code_counts,
  total_media_titles = 10,
  table_min_count = 5,
  table_min_prop = 0.3,
  output_type = "kable"
)

# Generate a plot of proportions
res <- set_saturation(
  code_counts,
  total_media_titles = 10,
  plot = TRUE,
  plot_metric = "prop"
)
res$plot

# Plot both count and proportion using dual y-axes
res <- set_saturation(
  code_counts,
  total_media_titles = 10,
  plot = TRUE,
  plot_metric = "both",
  fill_color = "darkgreen"
)
res$plot
```

view_excerpts *View Qualitative Excerpts by Code*

Description

Displays qualitative excerpts interactively in a searchable, filterable data table. Each row represents an excerpt associated with one or more qualitative codes. Code columns are automatically detected as those starting with "c_", and their variable labels (if available) are used as readable code names.

This function is primarily designed for exploring and reviewing coded qualitative data, allowing users to filter by code and quickly browse the corresponding excerpts.

Usage

```
view_excerpts(data)
```

Arguments

data	A data frame containing at least one text column named excerpt and one or more logical code columns prefixed with "c_". Each logical column represents whether a code was applied (TRUE/FALSE).

Details

- Variable labels are extracted from the "label" attribute of each code column (e.g., assigned via haven::labelled or attr(x, "label") <- "Label").
- Only excerpts where a code is marked as TRUE are displayed.
- The table uses custom styling with a purple header and automatic text wrapping.

Value

A DT::datatable() object that displays:

- **Code:** readable code label or variable name
- **Excerpt:** associated qualitative text

The output table includes:

- A dropdown filter for selecting specific codes
- Search boxes for column-wise filtering
- Responsive column widths and formatted text wrapping

Examples

```
library(dplyr)

df <- tibble::tibble(
  excerpt = c(
    "I felt supported by my peers.",
    "Teachers really listened to us.",
    "I learned a lot about myself."
  ),
  c_support = c(TRUE, TRUE, FALSE),
  c_growth = c(FALSE, FALSE, TRUE)
)
attr(df$c_support, "label") <- "Peer/Teacher Support"
attr(df$c_growth, "label") <- "Personal Growth"

# View excerpts interactively
if (interactive()) view_excerpts(df)
```

wordcloud *Generate a word cloud for excerpts by code*

Description

Creates a word cloud of words from all excerpts where a given code is applied. Common English stop words, user-supplied stop words, and punctuation are removed.

Usage

```
wordcloud(data, code, max_words = 100, custom_stopwords = NULL)
```

Arguments

`data` A data.frame or tibble containing at least one `excerpt` column and one or more code columns starting with `"c_"`.

`code` A string giving the name of the code column to filter on (e.g. "c_belonging").

`max_words` Maximum number of words to display in the word cloud (default = 100).

`custom_stopwords` A character vector of additional stop words to remove (default = `NULL`).

Value

An interactive word cloud (from **wordcloud2**).

Examples

```
library(dplyr)
df <- tibble::tibble(
  excerpt = c(
    "I felt connected to peers and friends.",
    "We should normalize conversations about mental health.",
    "My teachers helped me belong at school.",
    "I am comfortable talking about suicide prevention."
  ),
  c_belonging = c(TRUE, FALSE, TRUE, FALSE),
  c_destigmatization = c(FALSE, TRUE, FALSE, FALSE)
)

# Word cloud for belonging excerpts
wordcloud(df, "c_belonging")

# With custom stop words
wordcloud(df, "c_belonging", custom_stopwords = c("connected", "school"))
```

Index