Package 'SemNetCleaner'

June 9, 2020

Title An Automated Cleaning Tool for Semantic and Linguistic Data

Version 1.2.0

Date 2020-06-08

Maintainer Alexander P. Christensen <alexpaulchristensen@gmail.com>

Description Implements several functions that automates the cleaning and spell-checking of text data. Also converges, finalizes, removes plurals and continuous strings, and puts text data in binary format for semantic network analysis. Uses the 'SemNetDictionaries' package to make the cleaning process more accurate, efficient, and reproducible.

License GPL (>= 3.0)

URL https://github.com/AlexChristensen/SemNetCleaner

BugReports https://github.com/AlexChristensen/SemNetCleaner/issues

NeedsCompilation no

Encoding UTF-8

LazyData true

Depends R (>= 3.6.0), SemNetDictionaries (>= 0.1.5)

Imports stringdist, hunspell, searcher, tcltk, foreign, readxl, R.matlab, stringi

Suggests knitr, rmarkdown, htmlTable

VignetteBuilder knitr

RoxygenNote 7.1.0

Author Alexander P. Christensen [aut, cre] (<https://orcid.org/0000-0002-9798-7037>)

Repository CRAN

Date/Publication 2020-06-09 16:20:02 UTC

R topics documented:

`SemNetCleaner-package` *SemNetCleaner–package*

Description

Implements several functions that automates the cleaning and spell-checking of text data. Also converges, finalizes, removes plurals and continuous strings, and puts text data in binary format for semantic network analysis. Uses the `SemNetDictionaries` package to make the cleaning process more accurate, efficient, and reproducible.

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

See Also

Useful links:

- `https://github.com/AlexChristensen/SemNetCleaner`
- Report bugs at `https://github.com/AlexChristensen/SemNetCleaner/issues`

`best.guess` *Makes Best Guess for Spelling Correction*

Description

A wrapper function for the best guess of a spelling mistake based on the letters, the ordering of those letters, and the potential for letters to be interchanged. The Damerau-Levenshtein distance is used to guide inferences into what word the participant was trying to spell from a dictionary (see `SemNetDictionaries`)

Usage

```
best.guess(word, full.dictionary, dictionary = NULL, tolerance = 1)
```

Arguments

`word`	Character. A word to get best guess spelling options from dictionary
`full.dictionary`	Character vector. The dictionary to search for best guesses in. See `SemNetDictionaries`
`dictionary`	Character. A dictionary from `SemNetDictionaries` for monikers (enhances guessing)
`tolerance`	Numeric. The distance tolerance set for automatic spell-correction purposes. This function uses the function `stringdist` to compute the Damerau-Levenshtein distance, which is used to determine potential best guesses Unique words (i.e., $n = 1$) that are within the (distance) tolerance are automatically output as best guess responses. This default is based on Damerau's (1964) proclamation that more than 80% of all human misspellings can be expressed by a single error (e.g., insertion, deletion, substitution, and transposition). If there is more than one word that is within or below the distance tolerance, then these will be provided as potential options. The recommended and default distance tolerance is `tolerance = 1`, which only spell corrects a word if there is only one word with a DL distance of 1.

Value

The best guess(es) of the word

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, *7*, 171-176.

Examples

```
# Misspelled "bombay"
best.guess("bomba", full.dictionary = SemNetDictionaries::animals.dictionary)
```

---

bin2resp *Binary Responses to Character Responses*

Description

Converts the binary response matrix into characters for each participant

Usage

```
bin2resp(rmat, to.data.frame = FALSE)
```

Arguments

rmat	Binary matrix. A binarized response matrix of verbal fluency or linguistic data
to.data.frame	Boolean. Should output be a data frame where participants are columns? Defaults to FALSE. Set to TRUE to convert output to data frame

Value

A list containing objects for each participant and their responses

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Toy example
raw <- open.animals[c(1:10),-c(1:3)]

if(interactive())
{
  # Clean and prepocess data
  clean <- textcleaner(open.animals[,-c(1:2)], partBY = "row", dictionary = "animals")

  # Change binary response matrix to word response matrix
  charmat <- bin2resp(clean$responses$binary)
}
```

---

correct.changes *Correct Changes from* textcleaner

Description

A function that corrects changes that were made automatically by textcleaner

Usage

```
correct.changes(textcleaner.obj, changes)
```

Arguments

textcleaner.obj	Object from textcleaner
changes	Matrix. A matrix with changes made the textcleaner object $spellcheck$automated

Value

This function returns the corrected lists from textcleaners:

`binary` A matrix of responses where each row represents a participant and each column represents a unique response. A response that a participant has provided is a '1' and a response that a participant has not provided is a '0'

`responses` A list containing two objects:

- clean A response matrix that has been spell-checked and de-pluralized with duplicates removed. This can be used as a final dataset for analyses (e.g., fluency of responses)
- original The original response matrix that has had white spaces before and after words response. Also converts all upper-case letters to lower case

`spellcheck` A list containing three objects:

- `full` All responses regardless of spell-checking changes
- `auto` Only the incorrect responses that were changed during spell-check

`removed` A list containing two objects:

- `rows` Identifies removed participants by their row (or column) location in the original data file
- `ids` Identifies removed participants by their ID (see argument `data`)

`partChanges` A list where each participant is a list index with each response that was been changed. Participants are identified by their ID (see argument `data`). This can be used to replicate the cleaning process and to keep track of changes more generally. Participants with `NA` did not have any changes from their original data and participants with missing data are removed (see `removed$ids`)

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Toy example
raw <- open.animals[c(1:10),-c(1:3)]

if(interactive())
{
    #Full test
    clean <- textcleaner(open.animals[,-c(1,2)], partBY = "row", dictionary = "animals")
}
```

letter.freq *Letter Frequencies Based on 40,000 Words*

Description

A vector corresponding the frequency of letters across 40,000 words. Retrieved from: http://pi.math.cornell.edu/~mec/2003-2004/cryptography/subs/frequencies.html

Usage

```
data(letter.freq)
```

Format

letter.freq (26-element numeric vector)

Examples

```
data("letter.freq")
```

open.animals *Openness and Verbal Fluency*

Description

Raw Animals verbal fluency data (n = 516) from Christensen et al. (2018).

Usage

```
data(open.animals)
```

Format

open.animals (matrix 516 x 38)

Details

First column is a grouping variable (`"Group"`) with `1` corresponding to low openness to experience and `2` to high openness to experience

Second column is the latent variable of openness to experience with Intellect items removed (see Christensen et al., 2018 for more details).

Third column is the ID variable for each participant.

Columns 4-38 are raw fluency data.

References

Christensen, A. P., Kenett, Y. N., Cotter, K. N., Beaty, R. E., & Silvia, P. J. (2018). Remotely close associations: Openness to experience and semantic memory structure. *European Journal of Personality*, *32*, 480-492. doi:10.1002/per.2157

Examples

```
data("open.animals")
```

`pluralize` *Converts Words to their Plural Form*

Description

A function to change words to their plural form. The rules for converting words to their plural forms are based on the grammar rules found here: https://www.grammarly.com/blog/plural-nouns/. This function handles most special cases and some irregular cases (see examples) but caution is necessary. If no plural form is identified, then the original word is returned.

Usage

```
pluralize(word)
```

Arguments

`word`	A word

Value

Returns the word in singular form, unless a plural form could not be found (then the original word is returned)

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Handles any prototypical cases
"dogs"
pluralize("dog")

"foxes"
pluralize("fox")

"wolves"
pluralize("wolf")
```

```
"octopi"
pluralize("octopus")

"taxa"
pluralize("taxon")

# And most special cases:
"wives"
pluralize("wife")

"roofs"
pluralize("roof")

"photos"
pluralize("photo")

# And some irregular cases:
"children"
pluralize("child")

"teeth"
pluralize("tooth")

"mice"
pluralize("mouse")
```

`qwerty.dist` *QWERTY Distance for Same Length Words*

Description

Computes QWERTY Distance for words that have the same number of characters. Distance is computed based on the number of keys a character is away from another character on a QWERTY keyboard

Usage

```
qwerty.dist(wordA, wordB)
```

Arguments

`wordA`	Character vector. Word to be compared
`wordB`	Character vector. Word to be compared

Value

Numeric value for distance between `wordA` and `wordB`

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
#Identical values for Damerau-Levenshtein
stringdist::stringdist("big", "pig", method="dl")

stringdist::stringdist("big", "bug", method="dl")

#Different distances for QWERTY
qwerty.dist("big", "pig")

qwerty.dist("big", "bug") # Probably meant to type "bug"
```

---

read.data *Read in Common Data File Extensions*

Description

A single function to read in common data file extensions. Note that this function is specialized for reading in text data in the format necessary for functions in SemNetCleaner

File extensions supported:

- .Rdata
- .rds
- .csv
- .xlsx
- .xls
- .sav
- .txt
- .mat

Usage

```
read.data(file = file.choose(), header = TRUE, sep = ",", ...)
```

Arguments

file	Character. A path to the file to load. Defaults to interactive file selection using file.choose
header	Boolean. A logical value indicating whether the file contains the names of the variables as its first line. If missing, the value is determined from the file format: header is set to TRUE if and only if the first row contains one fewer field than the number of columns

sep — Character. The field separator character. Values on each line of the file are separated by this character. If sep = "" (the default for `read.table`) the separator is a 'white space', that is one or more spaces, tabs, newlines or carriage returns

... — Additional arguments. Allows for additional arguments to be passed onto the respective read functions. See documentation in the list below:

- .Rdata `load`
- .rds `readRDS`
- .csv `read.table`
- .xlsx `read_excel`
- .xls `read_excel`
- .sav `read.spss`
- .txt `read.table`
- .mat `readMat`

Value

A data frame containing a representation of the data in the file. If file extension is ".Rdata", then data will be read to the global environment

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

R Core Team

R Core Team (2019). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

readxl

Hadley Wickham and Jennifer Bryan (2019). readxl: Read Excel Files. R package version 1.3.1. https://CRAN.R-project.org/package=readxl

R.matlab

Henrik Bengtsson (2018). R.matlab: Read and Write MAT Files and Call MATLAB from Within R. R package version 3.6.2. https://CRAN.R-project.org/package=R.matlab

Examples

```
# Use this example for your data
if(interactive())
{read.data()}

# Example for CRAN tests
## Create test data
test1 <- c(1:5, "6,7", "8,9,10")

## Path to temporary file
tf <- tempfile()
```

```
## Create test file
writeLines(test1, tf)

## Read in data
read.data(tf)

# See documentation of respective R functions for specific examples
```

---

resp2bin *Responses to binary matrix*

Description

Converts the response matrix to binary response matrix

Usage

```
resp2bin(resp)
```

Arguments

resp	Response matrix. A response matrix of verbal fluency or linguistic data

Value

A list containing objects for each participant and their responses

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Toy example
raw <- open.animals[c(1:10),-c(1:3)]

if(interactive())
{
  # Clean and prepocess data
  clean <- textcleaner(open.animals[,-c(1:2)], partBY = "row", dictionary = "animals")

  # Change response matrix to binary response matrix
  binmat <- resp2bin(clean$responses$corrected)
}
```

`singularize` *Converts Words to their Singular Form*

Description

A function to change words to their singular form. The rules for converting words to their singular forms are based on the ***inverse*** of the grammar rules found here: https://www.grammarly.com/blog/plural-nouns/. This function handles most special cases and some irregular cases (see examples) but caution is necessary. If no singular form is identified, then the original word is returned.

Usage

```
singularize(word)
```

Arguments

`word`	Character. A word

Value

Returns the word in singular form, unless a singular form could not be found (then the original word is returned)

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Handles any prototypical cases
# "dog"
singularize("dogs")

# "fox"
singularize("foxes")

# "wolf"
singularize("wolves")

# "octopus"
singularize("octopi")

# "taxon"
singularize("taxa")

# And most special cases:
# "wife"
singularize("wives")
```

```
# "fez"
singularize("fezzes")

# "roof"
singularize("roofs")

# "photo"
singularize("photos")

# And some irregular cases:
# "child"
singularize("children")

# "tooth"
singularize("teeth")

# "mouse"
singularize("mice")
```

`textcleaner` *Text Cleaner*

Description

An automated cleaning function for spell-checking, de-pluralizing, removing duplicates, and binarizing text data

Usage

```
textcleaner(
  data = NULL,
  miss = 99,
  partBY = c("row", "col"),
  dictionary = NULL,
  continue = NULL,
  walkthrough = NULL
)
```

Arguments

`data`	Matrix or data frame. A dataset of text data. Participant IDs will be automatically identified if they are included. If no IDs are provided, then their order in the corresponding row (or column is used). A message will notify the user how IDs were assigned
`miss`	Numeric or character. Value for missing data. Defaults to `99`
`partBY`	Character. Are participants by row or column? Set to `"row"` for by row. Set to `"col"` for by column

`dictionary` Character vector. Can be a vector of a corpus or any text for comparison. Dictionary to be used for more efficient text cleaning. Defaults to `NULL`, which will use `general.dictionary`
Use `dictionaries()` or `find.dictionaries()` for more options (See `SemNetDictionaries` for more details)

`continue` List. A result previously unfinished that still needs to be completed. Allows you to continue to manually spell-check their data after you've closed or errored out. Defaults to `NULL`

`walkthrough` Boolean. Whether a walkthrough should be provided (recommended for first time users). Defaults to `NULL`, which will ask whether you would like a walkthrough. Set to `TRUE` to do the walkthrough. Set to `FALSE` to skip the walkthrough

Value

This function returns a list containing the following objects:

`binary` A matrix of responses where each row represents a participant and each column represents a unique response. A response that a participant has provided is a '`1`' and a response that a participant has not provided is a '`0`'

`responses` A list containing two objects:

- clean A response matrix that has been spell-checked and de-pluralized with duplicates removed. This can be used as a final dataset for analyses (e.g., fluency of responses)
- original The original response matrix that has had white spaces before and after words response. Also converts all upper-case letters to lower case

`spellcheck` A list containing three objects:

- `full` All responses regardless of spell-checking changes
- `auto` Only the incorrect responses that were changed during spell-check

`removed` A list containing two objects:

- `rows` Identifies removed participants by their row (or column) location in the original data file
- `ids` Identifies removed participants by their ID (see argument `data`)

`partChanges` A list where each participant is a list index with each response that was been changed. Participants are identified by their ID (see argument `data`). This can be used to replicate the cleaning process and to keep track of changes more generally. Participants with `NA` did not have any changes from their original data and participants with missing data are removed (see `removed$ids`)

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Hornik, K., & Murdoch, D. (2010). Watch Your Spelling!. *The R Journal*, *3*, 22-28. doi:10.32614/RJ-2011-014

Examples

```
# Toy example
raw <- open.animals[c(1:10),-c(1:3)]

if(interactive())
{
    #Full test
    clean <- textcleaner(open.animals[,-c(1,2)], partBY = "row", dictionary = "animals")
}
```

Index