

Package: ncodeR (via r-universe)

September 15, 2026

Title Techniques for Automated Classifiers

Type Package

Author Cody L Marquart [aut, cre]
(<https://orcid.org/0000-0002-3387-6792>), Zachari Swiecki
[aut], Brendan Eagan [aut], David Williamson Shaffer [aut]

Maintainer Cody L Marquart <cody.marquart@wisc.edu>

Version 0.2.1.1

Description A set of techniques that can be used to develop, validate,
and implement automated classifiers. A powerful tool for
transforming raw data into meaningful information, 'ncodeR'
(Shaffer, D. W. (2017) Quantitative Ethnography. ISBN:
0578191687) is designed specifically for working with big data:
large document collections, logfiles, and other text data.

LazyData TRUE

BugReports <https://gitlab.com/epistemic-analytics/qe-packages/ncoder/issues>

Depends R (>= 3.0.0)

License GPL-3 | file LICENSE

Imports R6, rhoR, cli

Suggests testthat, magrittr, tm, knitr, rmarkdown

RoxygenNote 7.1.1

Repository <https://epistemic-analytics.r-universe.dev>

Date/Publication 2021-05-21 20:08:29 UTC

RemoteUrl <https://gitlab.com/epistemic-analytics/qe-packages/ncoder>

RemoteRef HEAD

RemoteSha 2a9e62eb24a5d2e2aa9872a4f525c9532fe9bacf

Contents

as.data.frame.Code	2
as.data.frame.CodeSet	3

autocode	4
Code	4
code.set	7
CodeSet	8
create.code	9
differences	10
expression.match	11
getHandSetIndices	11
getHandSetIndices2	12
handcode	12
ncodeR	13
old_test	14
print.summary.Code	14
print.summary.CodeSet	15
print.summary.TestList	16
RegexCode	16
resolve	18
RS.data	19
summary.Code	19
summary.CodeSet	20
summary.TestList	21
test	21
unseen_words	22
Index	23

as.data.frame.Code	<i>Code to data.frame</i>
--------------------	---------------------------

Description

Code to data.frame

Usage

```
## S3 method for class 'Code'  
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	Code object to convert
row.names	NULL or a character vector giving the row names for the data frame. Missing values are not allowed.
optional	logical. If TRUE, setting row names and converting column names
...	additional arguments to be passed to or from methods

Value

data.frame

Examples

```
data(RS.data)
rs = RS.data
newcode = create.code(name = "Data", expressions = c("number", "data"), excerpts = rs$text)
as.data.frame(newcode)
```

as.data.frame.CodeSet *CodeSet to data.frame*

Description

CodeSet to data.frame

Usage

```
## S3 method for class 'CodeSet'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	CodeSet to convert
row.names	NULL or a character vector giving the row names for the data frame. Missing values are not allowed.
optional	logical. If TRUE, setting row names and converting column names
...	additional arguments to be passed to or from methods

Value

data.frame

Examples

```
data(RS.data)
rs = RS.data
newcode = create.code(name = "Data", expressions = c("number", "data"),
  excerpts = rs$text)
code.set = code.set("Demo RS CodeSet", "CodeSet made for the demo",
  excerpts = rs$text, codes = c(newcode))
as.data.frame(code.set)
```

autocode	<i>Match a list of expressions against some set of excerpts</i>
----------	---

Description

Autocodes all codes provided, either directly with code or as part of a provided codeset

Usage

```
autocode(  
  x = NULL,  
  expressions = NULL,  
  excerpts = NULL,  
  simplify = T,  
  mode = "all"  
)
```

Arguments

x	Object to autocode. Either a Code or CodeSet
expressions	Expressions to use for coding (optional)
excerpts	Excerpts to code
simplify	If TRUE, returns a data.frame, else returns a Code or CodeSet object
mode	Either all, training, or test representing the set of excerpts that should be recoded in the computerSet

Value

data.frame of is simplify = T (default), otherwise the Code or CodeSet object with updated computerSets

Code	<i>ncodeR Code object</i>
------	---------------------------

Description

ncodeR Code object
ncodeR Code object

Value

Code object Create Code

Public fields

call TBD
name TBD
definition TBD
codeSet TBD
testSet TBD
trainingSet TBD
computerSet TBD
ignoredSet TBD
examples TBD
excerpts TBD
holdoutExcerpts TBD
touchableExcerpts TBD
touchedIndices TBD
additionalExcerpts TBD
secondRaterSet TBD
statistics TBD
baserate TBD
baserateInflation TBD

Methods**Public methods:**

- [Code\\$new\(\)](#)
- [Code\\$process\(\)](#)
- [Code\\$kappa\(\)](#)
- [Code\\$differences\(\)](#)
- [Code\\$clearTestSet\(\)](#)
- [Code\\$concat\(\)](#)
- [Code\\$getValue\(\)](#)
- [Code\\$setValue\(\)](#)
- [Code\\$clone\(\)](#)

Method new():

Usage:

```
Code$new(  
  name,  
  definition,  
  testSet = NULL,  
  trainingSet = NULL,  
  computerSet = NULL,
```

```

    secondRaterSet = NULL,
    ignoredSet = c(),
    examples = NULL,
    excerpts = NULL,
    holdoutSize = 0.9,
    ...
)

```

Arguments:

```

name TBD
definition TBD
testSet TBD
trainingSet TBD
computerSet TBD
secondRaterSet TBD
ignoredSet TBD
examples TBD
excerpts TBD
holdoutSize TBD
... TBD

```

Returns: Code object Function to override

Method process():

Usage:

```
Code$process()
```

Returns: Default, error, unless overridden Kappa on Code

Method kappa():

Usage:

```
Code$kappa(which = c("training", "test"))
```

Arguments:

```
which TBD
```

Returns: double Differences in sets

Method differences():

Usage:

```
Code$differences(data = NULL, col1 = NULL, col2 = NULL, cols = NULL)
```

Arguments:

```

data TBD
col1 TBD
col2 TBD
cols TBD

```

Returns: vector of indices Clear TestSet

Method clearTestSet():*Usage:*

Code\$clearTestSet()

Returns: NULL Concatenate Expressions**Method** concat():*Usage:*

Code\$concat()

Returns: character of concatenated expressions Get Value**Method** getValue():*Usage:*

Code\$getValue(wh)

Arguments:

wh character

Returns: value stored at wh Set Value**Method** setValue():*Usage:*

Code\$setValue(wh, val)

Arguments:

wh character

val object

Method clone(): The objects of this class are cloneable with this method.*Usage:*

Code\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

code.setCreate CodeSet

Description

Create a new CodeSet object

Usage

code.set(title = "", description = "", excerpts = c(), codes = c())

Arguments

title	Title for the CodeSet
description	Description of the CodeSet
excerpts	Set of excerpts to use with the CodeSet
codes	Set of codes to attach to the CodeSet

Value

CodeSet object

Examples

```
data(RS.data)
rs = RS.data
code.set = code.set("Demo RS CodeSet", "CodeSet made for the demo", excerpts = rs$text, codes = c())
```

CodeSet	<i>CodeSet</i>
---------	----------------

Description

Object representing a set of codes

Value

CodeSet object Create CodeSet

Public fields

title	Title of the CodeSet
call	TBD
codes	list of Codes
description	String description of the set of codes to be included
excerpts	Character vector of text excerpts to code (optional)

Methods

Public methods:

- [CodeSet\\$new\(\)](#)
- [CodeSet\\$clone\(\)](#)

Method new():

```
Usage:
CodeSet$new(title = "ne", description = NULL, excerpts = c(), codes = c())
```

Arguments:
title TBD
description TBD
excerpts TBD
codes TBD
Returns: CodeSet

Method clone(): The objects of this class are cloneable with this method.

Usage:
CodeSet\$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.

Examples

```
data(RS.data)
rs = RS.data
code.set = code.set("Demo RS CodeSet", "CodeSet made for the demo", excerpts = rs$text, codes = c())
```

create.code	Create a code
-------------	---------------

Description

Create a code

Usage

```
create.code(
  name = "NewCode",
  definition = NULL,
  excerpts = NULL,
  type = "Regex",
  ...
)
```

Arguments

name	Name of the code
definition	Definition of the Code
excerpts	Character vectore of excerpts to use for Coding
type	Character string representing the type of code (Default: "Regex")
...	Additional parameters

Value

Code object

Examples

```
data(RS.data)
rs = RS.data

# Generate a Code
newcode = create.code(name = "Data", expressions = c("number","data"), excerpts = rs$text)
```

differences	<i>Find Differences</i>
-------------	-------------------------

Description

Find rows that differ within a data.frame or two vectors

Usage

```
differences(code = NULL, wh = "trainingSet", to = "computerSet")
```

Arguments

code	Code object to search for differences
wh	Set to use as the base comparison
to	Set to compare wh to

Value

logical vector representing indices that are coded differently
vector of indices representing differences

expression.match	<i>Expression Matching</i>
------------------	----------------------------

Description

Match a set of text excerpts against a set of regular expressions

Usage

```
expression.match(excerpts, expressions, names = list(NULL, "V1"))
```

Arguments

excerpts	Character vector to match against
expressions	Character vector of expressions
names	Character vector to use for dimension names

Value

Matrix representing matched expressions

getHandSetIndices	<i>Handset indices</i>
-------------------	------------------------

Description

Handset indices

Usage

```
getHandSetIndices(
  codeToUse,
  handSetLength = 20,
  handSetBaserate = 0.2,
  unseen = F
)
```

Arguments

codeToUse	[TBD]
handSetLength	[TBD]
handSetBaserate	[TBD]
unseen	[TBD]

getHandSetIndices2	<i>Get indices to code</i>
--------------------	----------------------------

Description

Get indices to code

Usage

```
getHandSetIndices2(  
  code,  
  handSetLength = 20,  
  handSetBaserate = 0.2,  
  unseen = FALSE,  
  this.set = NULL  
)
```

Arguments

code	Code object
handSetLength	Number of excerpts to put into the test set
handSetBaserate	Minimum number of positives that should be in the test set
unseen	[TBD]
this.set	[TBD]

Value

Code object with an updated test set and computer set

handcode	<i>Handcode Code</i>
----------	----------------------

Description

Handcode a set of excerpts using a vector of expressions

Usage

```

handcode(
  code = NULL,
  excerpts = NULL,
  expressions = NULL,
  n = ifelse(is.null(this.set), 10, length(this.set)),
  baserate = 0.2,
  unseen = FALSE,
  this.set = NULL,
  results = NULL
)

```

Arguments

code	Code object to handcode
excerpts	Excerpts to code (optional)
expressions	Expressions to code with (options)
n	Number of excerpts to handcode
baserate	Value between 0 and 1, inflates the baserate chosen excerpts to code, ensuring the number of positive at least equal to $n * \text{baserate}$
unseen	Logical or number Indicating additional excerpts with unseen words should be added. If TRUE (default), two words added or by 'number'
this.set	[TBD]
results	[TBD]

Value

Code

ncodeR

ncodeR for qualitative coding

Description

ncodeR is used for generating codes and coding datasets

old_test	<i>Calculate statistics</i>
----------	-----------------------------

Description

Run tests (kappa, rho) on the given Code

Usage

```
old_test(
  code,
  kappaThreshold = 0.65,
  baserateInflation = 0.2,
  type = c("training", "test")
)
```

Arguments

code	Code object to test
kappaThreshold	Threshold used for calculating rhoR::rho
baserateInflation	inflation rate to use when sampling handsets
type	vector indicating which stats should be calculated

Value

Code object with updated statistics property

print.summary.Code	<i>Print a Code summary</i>
--------------------	-----------------------------

Description

Print a Code summary

Usage

```
## S3 method for class 'summary.Code'
print(x, ...)
```

Arguments

x	list from summary()
...	Additional parameters

Value

Prints code summary

Examples

```
data(RS.data)
rs = RS.data
newcode = create.code(name = "Data",
  expressions = c("number","data"), excerpts = rs$text)
summary(newcode)
```

```
print.summary.CodeSet Print the summary of a CodeSet
```

Description

Print the summary of a CodeSet

Usage

```
## S3 method for class 'summary.CodeSet'
print(x, ...)
```

Arguments

x	Summary of a CodeSet
...	Additional parameters

Value

prints summary

Examples

```
data(RS.data)
rs = RS.data

newcode = create.code(name = "Data",
  expressions = c("number","data"), excerpts = rs$text)
code.set = code.set("Demo RS CodeSet", "CodeSet made for the demo",
  excerpts = rs$text, codes = c(newcode))
summary(code.set)
```

```
print.summary.TestList
```

Print a TestList summary

Description

Print a TestList summary

Usage

```
## S3 method for class 'summary.TestList'
print(x, ...)
```

Arguments

x	list from summary()
...	Additional parameters

Value

prints summary

Examples

```
data(RS.data)
rs = RS.data
newcode <- create.code("Data", expressions = c("number","data"), excerpts = rs$text)
newcode <- handcode(newcode, this.set = 10:15, results = 0)
newcode = test(code = newcode, kappa_threshold = 0.65)
summary(newcode$statistics)
```

RegexCode

RegexCode

Description

Creates an object for Regular Expression coding. No need to call this directly, create.code is a nice wrapper around this and any other types of Codes

Value

RegexCode object Create RegexCode

Super class

`ncodeR::Code` -> RegexCode

Public fields

expressions Character vector of regular expressions
metadata TBD

Methods**Public methods:**

- [RegexCode\\$new\(\)](#)
- [RegexCode\\$process\(\)](#)
- [RegexCode\\$add\(\)](#)
- [RegexCode\\$remove\(\)](#)
- [RegexCode\\$concat\(\)](#)
- [RegexCode\\$clone\(\)](#)

Method new():

Usage:

RegexCode\$new(name, definition, ..., excerpts = NULL, expressions = NULL)

Arguments:

name TBD

definition TBD

... TBD

excerpts TBD

expressions TBD

Returns: Process Code

Method process():

Usage:

RegexCode\$process(excerpts = self\$excerpts)

Arguments:

excerpts Options excerpts to code

Returns: Code results Add word

Method add():

Usage:

RegexCode\$add(word)

Arguments:

word To add to expressions

Returns: NULL Remove word

Method remove():

Usage:

RegexCode\$remove(word)

Arguments:
word to remove

Returns: NULL Concatenate expressions

Method concat():

Usage:
RegexCode\$concat()

Returns: character of concatentated expressions

Method clone(): The objects of this class are cloneable with this method.

Usage:
RegexCode\$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Examples

```
data(RS.data)
rs = RS.data

# Generate a Code
newcode = RegexCode$new(name = "New Code", definition = "Some definition",
  excerpts = rs$text, expressions = c("number","data"))
```

resolve	<i>Resolve differences</i>
---------	----------------------------

Description

Resolve differing results

Usage

```
resolve(
  code = NULL,
  trainingSet = NULL,
  computerSet = NULL,
  expressions = NULL,
  excerpts = NULL,
  ignored = NULL
)
```

Arguments

<code>code</code>	Code to resolve coding differences
<code>trainingSet</code>	Optionally provide a trainingSet, default: <code>code\$trainingSet</code>
<code>computerSet</code>	Optionally provide a computerSet, default: <code>code\$computerSet</code>
<code>expressions</code>	Optionally provide a set of expressions, default: <code>code\$expressions</code>
<code>excerpts</code>	Optionally provide a set of excerpts, default: <code>code\$excerpts</code>
<code>ignored</code>	Optionally provide a set of excerpts to ignore during the resolve cycle loop

RS.data

*Rescushell Chat Data***Description**

A dataset containing sample chat data from the Rescushell Virtual Internship

Usage

```
RS.data
```

Format

An object of class `data.frame` with 3824 rows and 20 columns.

summary.Code

*Obtain summary of a Code object***Description**

Obtain summary of a Code object

Usage

```
## S3 method for class 'Code'
summary(object, ...)
```

Arguments

<code>object</code>	Code to summarize
<code>...</code>	Additional parameters

Value

List of Code summary

Examples

```
data(RS.data)
rs = RS.data
newcode = create.code(name = "Data",
  expressions = c("number","data"), excerpts = rs$text)
summary(newcode)
```

summary.CodeSet	<i>Obtain a summary of the CodeSet</i>
-----------------	--

Description

Obtain a summary of the CodeSet

Usage

```
## S3 method for class 'CodeSet'
summary(object, ...)
```

Arguments

object	CodeSet object
...	Additional parameters

Value

list containing description and Code summaries

Examples

```
data(RS.data)
rs = RS.data

newcode = create.code(name = "Data",
  expressions = c("number","data"), excerpts = rs$text)
code.set = code.set("Demo RS CodeSet", "CodeSet made for the demo",
  excerpts = rs$text, codes = c(newcode))
summary(code.set)
```

summary.TestList	<i>Obtain a summary of a Code's test results</i>
------------------	--

Description

Obtain a summary of a Code's test results

Usage

```
## S3 method for class 'TestList'
summary(object, ...)
```

Arguments

object	TestList object of Code
...	Additional parameters

Value

list of Test summary

Examples

```
data(RS.data)
rs = RS.data
newcode = create.code(name = "Data",
  expressions = c("number","data"), excerpts = rs$text)
newcode <- handcode(newcode, this.set = 10:15, results = 0)
newcode = test(code = newcode, kappa_threshold = 0.65)
summary(newcode$statistics)
```

test	<i>Test code</i>
------	------------------

Description

Test code

Usage

```
test(
  code,
  kappa_threshold = 0.65,
  baserate_inflation = 0.2,
  ignore_warning = FALSE,
  ...
)
```

Arguments

code	TBD
kappa_threshold	
	TBD
baserate_inflation	
	TBD
ignore_warning	logical, when FALSE (default) verifies test set size before executing test
...	TBD

Value

code object

unseen_words	<i>Unseen Words</i>
--------------	---------------------

Description

Unseen Words

Usage

```
unseen_words(  
  code,  
  unweighted = TRUE,  
  include_handcoded = FALSE,  
  exclude_matched_by = c("excerpts", "word")  
)
```

Arguments

code	Code object
unweighted	logical TRUE (default), binarized the document matrix, so multiple occurrences of single word in one line doesn't count multiple times
include_handcoded	
	logical FALSE (default), will not use words from the handcoded set. If TRUE, the handcoded set will be used
exclude_matched_by	
	character, either "excerpts" (default) or "word". "excerpts" will remove full excerpts that match an expression, "word" will remove words from WDM to not include in sum of excerpts

Value

Excerpts that contain unseen words

Index

* **datasets**

- RS.data, [19](#)
- as.data.frame.Code, [2](#)
- as.data.frame.CodeSet, [3](#)
- autocode, [4](#)
- Code, [4](#)
- code.set, [7](#)
- CodeSet, [8](#)
- create.code, [9](#)
- differences, [10](#)
- expression.match, [11](#)
- getHandSetIndices, [11](#)
- getHandSetIndices2, [12](#)
- handcode, [12](#)
- ncodeR, [13](#)
- ncodeR::Code, [16](#)
- old_test, [14](#)
- print.summary.Code, [14](#)
- print.summary.CodeSet, [15](#)
- print.summary.TestList, [16](#)
- RegexCode, [16](#)
- resolve, [18](#)
- RS.data, [19](#)
- summary.Code, [19](#)
- summary.CodeSet, [20](#)
- summary.TestList, [21](#)
- test, [21](#)
- unseen_words, [22](#)