

# Package: ncodeR (via r-universe)

August 26, 2024

**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** 0.2.0.1

**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, knitr, rmarkdown

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Date/Publication** 2019-11-19 18:30:02 UTC

**Repository** <https://codymarquart.r-universe.dev>

**RemoteUrl** <https://github.com/cran/ncodeR>

**RemoteRef** HEAD

**RemoteSha** 548f2aac6015dd7bbc92889a59651c4a01c645da

## Contents

as.data.frame.Code . . . . .	2
as.data.frame.CodeSet . . . . .	3
autocode . . . . .	4
code.set . . . . .	4
CodeSet . . . . .	5
create.code . . . . .	6
differences . . . . .	6
expression.match . . . . .	7
getHandSetIndices . . . . .	8
getHandSetIndices2 . . . . .	8
handcode . . . . .	9
ncode . . . . .	9
ncodeR . . . . .	10
old_test . . . . .	10
print.summary.Code . . . . .	10
print.summary.CodeSet . . . . .	11
print.summary.TestList . . . . .	12
RegexCode . . . . .	12
resolve . . . . .	13
RS.data . . . . .	14
summary.Code . . . . .	14
summary.CodeSet . . . . .	15
summary.TestList . . . . .	15
test . . . . .	16
<b>Index</b>	<b>17</b>

---

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

---

### Description

Title

### 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 *Title*


---

**Description**

Title

**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.set	<i>Create CodeSet</i>
----------	-----------------------

---

### 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

*CodeSet*

---

**Description**

Object representing a set of codes

**Usage**

CodeSet

**Format**

An object of class R6ClassGenerator of length 24.

**Value**

CodeSet object

CodeSet

**Fields**

title Title of the CodeSet

description String description of the set of codes to be included

excerpts Character vector of text excerpts to code (optional)

expressions Codes to include in the CodeSet (optional)

**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*Find Differences*

---

**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

**Details**

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

**Value**

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

---

expression.match      *Expression Matching*

---

**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`      *Handset indices*

---

**Description**

Handset indices

**Usage**

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

**Arguments**

<code>codeToUse</code>	[TBD]
<code>handSetLength</code>	[TBD]
<code>handSetBaserate</code>	[TBD]
<code>unseen</code>	[TBD]

---

`getHandSetIndices2`      *Get indices to code*

---

**Description**

Get indices to code

**Usage**

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

**Arguments**

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

**Value**

Code object with an updated test set and computer set



---

handcode	<i>Handcode excerpts</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 = F, 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]

**Details**

Handcode a set of excerpts using a vector of expressions

**Value**

Code

---

ncode	<i>Wrapper for the entire coding process</i>
-------	--

---

**Description**

Wrapper for the entire coding process

**Usage**

```
ncode()
```

---

ncodeR	<i>ncodeR for qualitative coding</i>
--------	--------------------------------------

---

**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

**Usage**

```
RegexCode
```

**Format**

An object of class R6ClassGenerator of length 24.

**Value**

RegexCode object

**Fields**

name Name of the Code  
 definition Definition of the Code  
 excerpts Character vector of text excerpts to code  
 ... Additional parameters not specific to a RegexCode  
 expressions Character vector of regular expressions

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

---

RS.data	<i>Rescushell Chat Data</i>
---------	-----------------------------

---

**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	<i>Obtain summary of a Code object</i>
--------------	--

---

**Description**

Obtain summary of a Code object

**Usage**

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

**Arguments**

object	Code to summarize
...	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>Title</i>
------	--------------

---

**Description**

Title

**Usage**

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

**Arguments**

code                [TBD]  
 kappa\_threshold    [TBD]  
 baserate\_inflation [TBD]  
 ...                 [TBD]

**Value**

code object



# Index

## \* datasets

- CodeSet, [5](#)
- RegexCode, [12](#)
- RS.data, [14](#)

[as.data.frame.Code](#), [2](#)  
[as.data.frame.CodeSet](#), [3](#)  
[autocode](#), [4](#)

[code.set](#), [4](#)  
[CodeSet](#), [5](#)  
[create.code](#), [6](#)

[differences](#), [6](#)

[expression.match](#), [7](#)

[getHandSetIndices](#), [8](#)  
[getHandSetIndices2](#), [8](#)

[handcode](#), [9](#)

[ncode](#), [9](#)  
[ncodeR](#), [10](#)

[old\\_test](#), [10](#)

[print.summary.Code](#), [10](#)  
[print.summary.CodeSet](#), [11](#)  
[print.summary.TestList](#), [12](#)

[RegexCode](#), [12](#)  
[resolve](#), [13](#)  
[RS.data](#), [14](#)

[summary.Code](#), [14](#)  
[summary.CodeSet](#), [15](#)  
[summary.TestList](#), [15](#)

[test](#), [16](#)