Package 'mde'

October 13, 2022

Title Missing Data Explorer

Version 0.3.2

Description Correct identification and handling of missing data is one of the most important steps in any analysis. To aid this process, 'mde' provides a very easy to use yet robust framework to quickly get an idea of where the missing data
lies and therefore find the most appropriate action to take.
Graham WJ (2009) <doi:10.1146/annurev.psych.58.110405.085530>.

License GPL-3

Depends R(>= 3.6.0)

Imports dplyr(>= 1.0.0), tidyr(>= 1.0.3)

Encoding UTF-8

RoxygenNote 7.1.2

URL https://github.com/Nelson-Gon/mde

BugReports https://github.com/Nelson-Gon/mde/issues

Suggests knitr, rmarkdown, markdown, testthat

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Nelson Gonzabato [aut, cre]

Maintainer Nelson Gonzabato <gonzabato@hotmail.com>

Repository CRAN

Date/Publication 2022-02-10 12:10:06 UTC

R topics documented:

`all_na` *Checks that all values are NA*

Description

This is a helper function to check if all column/vector values are NA

Usage

```
all_na(x)
```

Arguments

x	A vector or data.frame column

Value

Boolean TRUE or FALSE depending on the nature of the column/vector

Examples

```
test <- data.frame(A=c(NA, 2), B= c(NA, NA))
all_na(test)
test_vec <- c("NA",NA,"nope")
test_numeric <- c(NA, 2)
all_na(test_vec)
all_na(test_numeric)
```

column_based_recode *Conditionally Recode NA values based on other Columns*

Description

Recode NA as based on Other Columns

Usage

```
column_based_recode(
  df,
  criterion = "all_na",
  values_from = NULL,
  values_to = NULL,
  value = 0,
  pattern_type = "contains",
  pattern = "Solar",
  case_sensitive = FALSE
)
```

Arguments

df	A data.frame object for which recoding is to be done.
criterion	Currently supports one of all_na or any_na to index rows that are either all NA or contain any NA.
values_from	Character. Name of column to get the original values from
values_to	Character New column name for the newly recoded values. Defaults to the same name if none is supplied.
value	The value to convert to ‘NA‘. We can for instance change "n/a" to ‘NA‘ or any other value.
pattern_type	One of contains’, ’starts_with’ or ’ends_with’.
pattern	A character pattern to match
case_sensitive	Defaults to FALSE. Patterns are case insensitive if TRUE

Value

A ‘data.frame‘ object with target ‘NA‘ values replaced.

Examples

```
df <- structure(list(id = 40:43, v1 = c(NA, 1L, 1L, 1L), v2 = c(NA, 1L, 1L, 1L),
v3 = c(NA, 2L, NA, 1L),
test = c(1L, 2L, 1L, 3L)), class = "data.frame", row.names = c(NA, -4L))
# recode test as 0 if all NA, return test otherwise
column_based_recode(df,values_from = "test", pattern_type = "starts_with", pattern="v")
```

custom_na_recode *Recode NA as another value using a function or a custom equation*

Description

Recode NA as another value using a function or a custom equation

Usage

```
custom_na_recode(
  df,
  func = "mean",
  grouping_cols = NULL,
  across_columns = NULL
)
```

Arguments

df	A valid R 'object' for which the percentage of missing values is required.
func	Function to use for the replacement e.g "mean". Defaults to mean.
grouping_cols	A character vector. If supplied, one can provide the columns by which to group the data.
across_columns	A character vector specifying across which columns recoding should be done #use all columns head(custom_na_recode(airquality,func="mean")) # use only a few columns head(custom_na_recode(airquality,func="mean",across_columns = c("Solar.R","Ozone"))) # use a function from another package #head(custom_na_recode(airquality, func=dplyr::lead)) some_data <- data.frame(ID=c("A1","A1","A1","A2","A2", "A2"), A=c(5,NA,0,8,3,4), B=c(10,0,0,NA,5,6),C=c(1,NA,NA,25,7,8)) # grouping head(custom_na_recode(some_data,func = "mean", grouping_cols = "ID", across_columns = c("C", "A"))) head(custom_na_recode(some_data,func = "mean", grouping_cols = "ID"))

dict_recode *Recode Missing Values Dictionary-Style*

Description

Recode Missing Values Dictionary-Style

Usage

```
dict_recode(
  df,
  use_func = "recode_na_as",
  pattern_type = "starts_with",
  patterns,
  values
)
```

Arguments

df	A data.frame object for which recoding is to be done.
use_func	Function to use for the recoding. One of the various 'recode_*' functions in package 'mde'.
pattern_type	One of contains', 'starts_with' or 'ends_with'.
patterns	A vector containing patterns to use for pattern_type
values	A vector containing values to match to the patterns vector

Value

A 'data.frame' object with replacements as required.

Examples

```
head(dict_recode(airquality, pattern_type="starts_with",
patterns = c("Solar", "Ozone"), values = c(190, 41),
use_func="recode_as_na"))
 head(dict_recode(airquality, pattern_type="starts_with",
patterns = c("Solar", "Ozone"), values = c(42, 420),
use_func="recode_na_as"))
```

---

drop_all_na *Drop columns for which all values are NA*

Description

Drop columns for which all values are NA

Usage

```
drop_all_na(df, grouping_cols = NULL)
```

Arguments

df	A valid R 'object' for which the percentage of missing values is required.
grouping_cols	A character vector. If supplied, one can provide the columns by which to group the data.

Examples

```
test <- data.frame(ID= c("A","A","B","A","B"), Vals = c(rep(NA,4),2))
test2 <- data.frame(ID= c("A","A","B","A","B"), Vals = rep(NA, 5))
# drop columns where all values are NA
drop_all_na(test2)
# drop NAs only if all are NA for a given group, drops group too.
drop_all_na(test, "ID")
```

`drop_na_at` *Drop missing values at columns that match a given pattern*

Description

Provides a simple yet efficient way to drop missing values("NA"s) at columns that match a given pattern.

Usage

```
drop_na_at(
  df,
  pattern_type = "contains",
  pattern = NULL,
  case_sensitive = FALSE,
  ...
)
```

Arguments

`df`	A data.frame object
`pattern_type`	One of "contains", "ends_with" or "starts_with"
`pattern`	The type of pattern to use when matching the pattern_type. The pattern is case sensitive
`case_sensitive`	Defaults to FALSE. Patterns are case insensitive if TRUE
`...`	Other params to other methods

Value

A data.frame object containing only columns that match the given pattern with the missing values removed.

Examples

```
head(drop_na_at(airquality,pattern_type = "starts_with","O"))
```

drop_na_if *Condition based dropping of columns with missing values*

Description

"drop_na_if" provides a simple way to drop columns with missing values if they meet certain criteria/conditions.

Usage

```
drop_na_if(
  df,
  sign = "gteq",
  percent_na = 50,
  keep_columns = NULL,
  grouping_cols = NULL,
  target_columns = NULL,
  ...
)
```

Arguments

df	A data.frame object
sign	Character. One of gteq,lteq,lt,gt or eq which refer to greater than(gt) or equal(eq) or less than(lt) or equal to(eq) respectively.
percent_na	The percentage to use when dropping columns with missing values
keep_columns	Columns that should be kept despite meeting the target percent_na criterion(criteria)
grouping_cols	For dropping groups that meet a target criterion of percent missingness.
target_columns	If working on grouped data, drop all columns that meet target or only a specific column.
...	Other arguments to "percent_missing"

Value

A data.frame object with columns that meet the target criteria dropped.

See Also

percent_missing

Examples

```
head(drop_na_if(airquality, percent_na = 24))
#drop columns that have less tan or equal to 4%
head(drop_na_if(airquality,sign="lteq", percent_na = 4))
# Drop all except with greater than ie equal to 4% missing but keep Ozone
head(drop_na_if(airquality, sign="gteq",percent_na = 4,
keep_columns = "Ozone"))
# Drop groups that meet a given criterion
grouped_drop <- structure(list(ID = c("A", "A", "B", "A", "B"), Vals = c(4, NA,
NA, NA, NA), Values = c(5, 6, 7, 8, NA)), row.names = c(NA, -5L),
 class = "data.frame")
 drop_na_if(grouped_drop,percent_na = 67,grouping_cols = "ID")
```

---

drop_row_if *Conditionally drop rows based on percent missingness*

Description

Conditionally drop rows based on percent missingness

Usage

```
drop_row_if(df, sign = "gt", type = "count", value = 20, as_percent = TRUE)
```

Arguments

df	A data.frame object
sign	Character. One of gteq,lteq,lt,gt or eq which refer to greater than(gt) or equal(eq) or less than(lt) or equal to(eq) respectively.
type	One of either count or percent. Defaults to count
value	Value to use for the drop.
as_percent	Logical. If set to TRUE, percent_na is treated as a percentage. Otherwise, decimals(fractions) are used.

Examples

```
head(drop_row_if(airquality,sign = "gteq",
type = "percent",value=16, as_percent = TRUE))
# should give the same output as above.
head(drop_row_if(airquality, sign="gteq", type="percent",value = 0.15, as_percent=FALSE))
# Drop based on NA counts
df <- data.frame(A=1:5, B=c(1,NA,NA,2, 3), C= c(1,NA,NA,2,3))
drop_row_if(df, type="count",value=2,sign="eq")
```

get_na_counts *Add columnwise/groupwise counts of missing values*

Description

This function takes a 'data.frame' object as an input and returns the corresponding 'NA' counts. 'NA' refers to R's builtin missing data holder.

Usage

```
get_na_counts(x, grouping_cols = NULL, exclude_cols = NULL)
```

Arguments

x	A valid R 'object' for which 'na_counts' are needed.
grouping_cols	A character vector. If supplied, one can provide the columns by which to group the data.
exclude_cols	Columns to exclude from the analysis.

Value

An object of the same type as 'x' showing the respective number of missing values. If grouped is set to 'TRUE', the results are returned by group.

Examples

```
get_na_counts(airquality)
# Grouped counts
test <- data.frame(Subject = c("A","A","B","B"), res = c(NA,1,2,3),
ID = c("1","1","2","2"))
get_na_counts(test,grouping_cols = c("ID", "Subject"))
```

get_na_means *Get mean missingness.*

Description

Get mean missingness.

Usage

```
get_na_means(x, as_percent = TRUE)
```

Arguments

x	A vector whose mean NA is required.
as_percent	Boolean? Report means as percents, defaults to TRUE.

Examples

```
get_na_means(airquality)
```

`na_counts` *Get NA counts for a given character, numeric, factor, etc.*

Description

Get NA counts for a given character, numeric, factor, etc.

Usage

```
na_counts(x)
```

Arguments

x	A vector whose number of missing values is to be determined.

Examples

```
na_counts(airquality$Ozone)
```

`na_summary` *An all-in-one missingness report*

Description

An all-in-one missingness report

Usage

```
na_summary(
  df,
  grouping_cols = NULL,
  sort_by = NULL,
  descending = FALSE,
  exclude_cols = NULL,
  pattern = NULL,
  pattern_type = NULL,
  regex_kind = "exclusion",
  round_to = NULL,
  reset_rownames = FALSE
)
```

Arguments

df	A valid R 'object' for which the percentage of missing values is required.
grouping_cols	A character vector. If supplied, one can provide the columns by which to group the data.
sort_by	One of counts or percents. This determines whether the results are sorted by counts or percentages.
descending	Logical. Should missing values be sorted in decreasing order ie largest to smallest? Defaults to FALSE.
exclude_cols	A character vector indicating columns to exclude when returning results.
pattern	Pattern to use for exclusion or inclusion. column inclusion criteria.
pattern_type	A regular expression type. One of "starts_with", "contains", or "regex". Defaults to NULL. Only use for selective inclusion.
regex_kind	One of inclusion or exclusion. Defaults to exclusion to exclude columns using regular expressions.
round_to	Number of places to round 2. Defaults to user digits option.
reset_rownames	Should the rownames be reset in the output? defaults to FALSE

Examples

```
na_summary(airquality)
# grouping
test2 <- data.frame(ID= c("A","A","B","A","B"),Vals = c(rep(NA,4),"No"),
ID2 = c("E","E","D","E","D"))
df <- data.frame(A=1:5,B=c(NA,NA,25,24,53), C=c(NA,1,2,3,4))

na_summary(test2,grouping_cols = c("ID","ID2"))
# sort summary
na_summary(airquality,sort_by = "percent_missing",descending = TRUE)
na_summary(airquality,sort_by = "percent_complete")
# Include only via a regular expression
na_summary(mtcars, pattern_type = "contains",
pattern = "mpg|disp|wt", regex_kind = "inclusion")
na_summary(airquality, pattern_type = "starts_with",
pattern = "ozone", regex_kind = "inclusion")
# exclusion via a regex
na_summary(airquality, pattern_type = "starts_with",
pattern = "oz|Sol", regex_kind = "exclusion")
# reset rownames when sorting by variable
na_summary(df,sort_by="variable",descending=TRUE, reset_rownames = TRUE)
```

---

percent_missing *Column-wise missingness percentages*

Description

A convenient way to obtain percent missingness column-wise.

Usage

```
percent_missing(df, grouping_cols = NULL, exclude_cols = NULL)
```

Arguments

df	A valid R 'object' for which the percentage of missing values is required.
grouping_cols	A character vector. If supplied, one can provide the columns by which to group the data.
exclude_cols	A character vector indicating columns to exclude when returning results.

Value

An object of the same class as x showing the percentage of missing values.

Examples

```
test <- data.frame(ID= c("A","B","A","B","A","B","A"),
Vals = c(NA,25,34,NA,67,NA,45))
percent_missing(test,grouping_cols = "ID")
percent_missing(airquality)
percent_missing(airquality,exclude_cols = c("Day","Temp"))
```

---

percent_na *percent missing but for vectors.*

Description

percent missing but for vectors.

Usage

```
percent_na(x)
```

Arguments

x	A vector whose mean NA is required.

Examples

```
percent_na(airquality$Ozone)
```

`recode_as_na` *Recode a value as NA*

Description

This provides a convenient way to convert a number/value that should indeed be an "NA" to "NA". In otherwords, it converts a value to R's recognized NA.

Usage

```
recode_as_na(
  df,
  value = NULL,
  subset_cols = NULL,
  pattern_type = NULL,
  pattern = NULL,
  case_sensitive = FALSE,
  ...
)
```

Arguments

`df`	A data.frame object for which recoding is to be done.
`value`	The value to convert to 'NA'. We can for instance change "n/a" to 'NA' or any other value.
`subset_cols`	An optional character vector to define columns for which changes are required.
`pattern_type`	One of contains', 'starts_with' or 'ends_with'.
`pattern`	A character pattern to match
`case_sensitive`	Defaults to FALSE. Patterns are case insensitive if TRUE
`...`	Other arguments to other functions

Value

An object of the same class as x with values changed to 'NA'.

Examples

```
head(recode_as_na(airquality,value=c(67,118),pattern_type="starts_with",pattern="S|O"))
head(recode_as_na(airquality,value=c(41),pattern_type="ends_with",pattern="e"))
head(recode_as_na(airquality, value=41,subset_cols="Ozone"))
```

recode_as_na_for *Recode Values as NA if they meet defined criteria*

Description

Recode Values as NA if they meet defined criteria

Usage

```
recode_as_na_for(df, criteria = "gt", value = 0, subset_cols = NULL)
```

Arguments

df	A data.frame object to manipulate
criteria	One of gt,gteq,lt,lteq to define greater than, greater than or equal to, less than or less than or equal to.
value	The value to convert to 'NA'. We can for instance change "n/a" to 'NA' or any other value.
subset_cols	An optional character vector for columns to manipulate.

Value

A data.frame object with the required changes.

Examples

```
recode_as_na_for(airquality,value=36, criteria = "gteq",
subset_cols = c("Ozone","Solar.R"))
```

recode_as_na_if *Conditionally change all column values to NA*

Description

Conditionally change all column values to NA

Usage

```
recode_as_na_if(df, sign = "gteq", percent_na = 50, keep_columns = NULL, ...)
```

Arguments

df	A data.frame object
sign	Character. One of gteq,lteq,lt,gt or eq which refer to greater than(gt) or equal(eq) or less than(lt) or equal to(eq) respectively.
percent_na	The percentage to use when dropping columns with missing values
keep_columns	Columns that should be kept despite meeting the target percent_na criterion(criteria)
...	Other arguments to "percent_missing"

Value

A ‘data.frame‘ with the target columns populated with ‘NA‘s.

Examples

```
head(recode_as_na_if(airquality, sign="gt", percent_na=20))
```

---

recode_as_na_str *Recode as NA based on string match*

Description

Recode as NA based on string match

Usage

```
recode_as_na_str(
  df,
  pattern_type = "ends_with",
  pattern = NULL,
  case_sensitive = FALSE,
  ...
)
```

Arguments

df	A data.frame object
pattern_type	One of contains’, ’starts_with’ or ’ends_with’.
pattern	A character pattern to match
case_sensitive	Defaults to FALSE. Patterns are case insensitive if TRUE
...	Other arguments to grepl

See Also

recode_as_na recode_as_na_if

Examples

```
partial_match <- data.frame(A=c("Hi","match_me","nope"), B=c(NA, "not_me","nah"))
# Replace all that end with "me" with NA
recode_as_na_str(partial_match,"ends_with","me")
# Do not recode, ie case-sensitive
recode_as_na_str(partial_match,"ends_with","ME", case_sensitive=TRUE)
```

`recode_as_value` *Recode a value as another value*

Description

This provides a convenient way to convert a number/value to another value.

Usage

```
recode_as_value(
  df,
  value = NULL,
  replacement_value = NULL,
  subset_cols = NULL,
  pattern_type = NULL,
  pattern = NULL,
  case_sensitive = FALSE,
  ...
)
```

Arguments

`df`	A data.frame object for which recoding is to be done.
`value`	The value/vector of values to convert.
`replacement_value`	New value.
`subset_cols`	An optional character vector to define columns for which changes are required.
`pattern_type`	One of contains', 'starts_with' or 'ends_with'.
`pattern`	A character pattern to match
`case_sensitive`	Defaults to FALSE. Patterns are case insensitive if TRUE
`...`	Other arguments to other functions

Value

An object of the same class as x with values changed to 'NA'.

Examples

```
head(recode_as_value(airquality,
value=c(67,118),replacement=NA, pattern_type="starts_with",pattern="S|O"))
```

`recode_helper` *Helper functions in package mde*

Description

Helper functions in package mde

Usage

```
recode_helper(
  x,
  pattern_type = NULL,
  pattern = NULL,
  original_value,
  new_value,
  case_sensitive = FALSE,
  ...
)
```

Arguments

`x`	A data.frame object
`pattern_type`	One of contains', 'starts_with' or 'ends_with'.
`pattern`	A character pattern to match
`original_value`	Value to replace
`new_value`	Replacement value.
`case_sensitive`	Defaults to FALSE. Patterns are case insensitive if TRUE
`...`	Other arguments to other functions

`recode_na_as` *Replace missing values with another value*

Description

This provides a convenient way to recode "NA" as another value for instance "NaN", "n/a" or any other value a user wishes to use.

Usage

```
recode_na_as(
  df,
  value = 0,
  subset_cols = NULL,
  pattern_type = NULL,
  pattern = NULL,
  case_sensitive = FALSE,
  ...
)
```

Arguments

df	A data.frame object for which recoding is to be done.
value	The value to convert to ‘NA‘. We can for instance change "n/a" to ‘NA‘ or any other value.
subset_cols	An optional character vector to define columns for which changes are required.
pattern_type	One of contains’, ’starts_with’ or ’ends_with’.
pattern	A character pattern to match
case_sensitive	Defaults to FALSE. Patterns are case insensitive if TRUE
...	Other arguments to other functions

Value

An object of the same type as x with NAs replaced with the desired value.

Examples

```
head(recode_na_as(airquality, "n/a"))
head(recode_na_as(airquality, subset_cols = "Ozone", value = "N/A"))
head(recode_na_as(airquality, value=0, pattern_type="starts_with",
pattern="Solar"))
```

---

recode_na_if *Recode NA as another value with some conditions*

Description

Recode NA as another value with some conditions

Usage

```
recode_na_if(df, grouping_cols = NULL, target_groups = NULL, replacement = 0)
```

Arguments

df	A data.frame object with missing values
grouping_cols	Character columns to use for grouping the data
target_groups	Character Recode NA as if and only if the grouping column is in this vector of values
replacement	Values to use to replace NAs for IDs that meet the requirements. Defaults to 0.

Examples

```
some_data <- data.frame(ID=c("A1","A2","A3", "A4"),
A=c(5,NA,0,8), B=c(10,0,0,1),C=c(1,NA,NA,25))
# Replace NAs with 0s only for IDs in A2 and A3
recode_na_if(some_data,"ID",c("A2","A3"),replacement=0)
```

---

recode_selectors *Helper functions in package mde*

Description

Helper functions in package mde

Usage

```
recode_selectors(
  x,
  column_check = TRUE,
  pattern_type = NULL,
  pattern = NULL,
  case_sensitive = FALSE,
  ...
)
```

Arguments

x	data.frame object
column_check	If TRUE, pattern search is performed columnwise. Defaults to FALSE.
pattern_type	One of contains', 'starts_with' or 'ends_with'.
pattern	A character pattern to match
case_sensitive	Defaults to FALSE. Patterns are case insensitive if TRUE
...	Other arguments to other functions

`sort_by_missingness` *Sort Variables according to missingness*

Description

Provides a useful way to sort the variables(columns) according to their missingness.

Usage

```
sort_by_missingness(df, sort_by = "counts", descending = FALSE, ...)
```

Arguments

`df`	A data.frame object
`sort_by`	One of counts or percents. This determines whether the results are sorted by counts or percentages.
`descending`	Logical. Should missing values be sorted in decreasing order ie largest to smallest? Defaults to FALSE.
`...`	Other arguments to specific functions. See "See also below"

Value

A 'data.frame' object sorted by number/percentage of missing values

See Also

`get_na_counts` `percent_missing`

Examples

```
sort_by_missingness(airquality, sort_by = "counts")
# sort by percents
sort_by_missingness(airquality, sort_by="percents")
# descending order
sort_by_missingness(airquality, descend = TRUE)
```

Index