

Package ‘IPDFFileCheck’

October 12, 2022

Type Package

Title Basic Functions to Check Readability, Consistency, and Content of an Individual Participant Data File

Version 0.7.5

Author Sheeja Manchira Krishnan

Maintainer Sheeja Manchira Krishnan <sheejamk@gmail.com>

Description Basic checks needed with an individual level participant data from randomised controlled trial. This checks files for existence, read access and individual columns for formats. The checks on format is currently implemented for gender and age formats.

Imports dplyr, testthat, lubridate, methods, eeptools, hash, kableExtra, gtsummary, effsize, stringr, tidyverse, tidyselct, gmodels, lmtest, zoo

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.2

Depends R (>= 3.6.0)

Suggests knitr, rmarkdown, covr

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Repository CRAN

Date/Publication 2022-02-01 08:00:10 UTC

R topics documented:

| | |
|---------------------------|---|
| calculate_age_from_dob | 2 |
| calculate_age_from_year | 3 |
| check_column_exists | 4 |
| check_col_pattern_colname | 5 |

| | |
|--|----|
| check_load_packages | 5 |
| cohensd | 6 |
| convert_date_numeric_stdform | 6 |
| convert_date_string_stdform | 7 |
| convert_to_number | 7 |
| descriptive_stats_col_excl_nrcode | 8 |
| get_colno_pattern_colname | 9 |
| get_columnno_fornames | 9 |
| get_contents_cols | 10 |
| get_effect_size | 10 |
| get_mode_from_vector | 11 |
| get_sem | 12 |
| get_summary_gtsummary | 12 |
| get_value_from_codes | 13 |
| keep_required_columns | 14 |
| present_mean_sd_rmna_text | 14 |
| represent_categorical_data_exclude_missing | 15 |
| represent_categorical_data_forsubgroups | 16 |
| represent_categorical_data_include_missing | 16 |
| represent_categorical_textdata | 17 |
| represent_numerical_data_forsubgroups | 18 |
| return_longitudinal_summary | 18 |
| return_subgroup_omitna | 19 |
| return_subgroup_withNA | 20 |
| test_age | 21 |
| test_columnnames | 21 |
| test_column_contents | 22 |
| test_data_numeric | 23 |
| test_data_numeric_norange | 23 |
| test_data_string | 24 |
| test_data_string_restriction | 25 |
| test_file_exist_read | 25 |
| test_gender | 26 |

Index **27**

calculate_age_from_dob

Function to calculate age from date of birth

Description

Function to calculate age from date of birth

Usage

```
calculate_age_from_dob(  
  data,  
  columnname,  
  enddatecol = NULL,  
  dateformat = "dmy",  
  nrcode = NA  
)
```

Arguments

| | |
|------------|---|
| data | a data frame |
| columnname | name of column corresponding to date of birth |
| enddatecol | column containing when to calculate the age to, default value is null, this means the age is calculated to the current date |
| dateformat | format of date e.g. dmy default is dmy |
| nrcode | non response code corresponding to date of birth |

Value

data if success error if failure

Examples

```
library(IPDFileCheck)  
this.df <- data.frame(c("1987-05-28", "1987-06-18"), c(1, 2),  
  stringsAsFactors = FALSE)  
colnames(this.df) <- c("dob", "num")  
calculate_age_from_dob(this.df, "dob", NULL, "ymd")
```

calculate_age_from_year

Function to calculate age from year of birth

Description

Function to calculate age from year of birth

Usage

```
calculate_age_from_year(data, columnname, endyearcol = NULL, nrcode = NA)
```

Arguments

| | |
|------------|---|
| data | a data frame |
| columnname | name of column corresponding to year of birth |
| endyearcol | name of column where the year is entered to calculate the age upto, by default its the current year |
| nrcode | non response code corresponding to date of birth |

Value

data, if success error if failure

Examples

```
this.data.frame <- data.frame(c(1951, 1980), c("John", "Dora"))
colnames(this.data.frame) <- c("yob", "name")
calculate_age_from_year(this.data.frame, "yob", NULL, NA)
```

check_column_exists *Function to check the given column exists*

Description

Function to check the given column exists

Usage

```
check_column_exists(column_name, data)
```

Arguments

| | |
|-------------|---------------|
| column_name | a column name |
| data | data frame |

Value

0 if success error if failure

Examples

```
check_column_exists("age", data.frame("Age" = c(21, 15),
"Name" = c("John", "Dora")))
```

```

check_col_pattern_colname
#####
Function to check if a given pattern is contained in the column names
of a data

```

Description

Function to check if a given pattern is contained in the column names of a data

Usage

check_col_pattern_colname(pattern, column_names)

Arguments

pattern a string that needs to be checked
column_names column names actually have

Value

TRUE, if success FALSE, if failure

Examples

check_col_pattern_colname("age", "female_age")

```

check_load_packages    Function to check the package is installed, if not install

```

Description

Function to check the package is installed, if not install

Usage

check_load_packages(pkg)

Arguments

pkg name of package(s)

Value

0, if packages cant be installed and loaded, else error

Examples

```
check_load_packages("dplyr")
```

| | |
|---------|---|
| cohensd | <i>Function to find the effect size Cohen's d</i> |
|---------|---|

Description

Function to find the effect size Cohen's d

Usage

```
cohensd(x, y)
```

Arguments

| | |
|----|----------------|
| x, | a vector |
| y, | another vector |

Value

cohens d estimated with 95

Examples

```
cohensd(c(1, 2, 3, 4), c(3, 4, 5, 6))
```

| | |
|------------------------------|--|
| convert_date_numeric_stdform | <i>Helper function to keep date formats in year-month-date</i> |
|------------------------------|--|

Description

Helper function to keep date formats in year-month-date

Usage

```
convert_date_numeric_stdform(column, index, orderby = "dmy")
```

Arguments

| | |
|---------|---|
| column | a data frame or a vector |
| index | those correspond to valid date in numeric form (omitting non response code or no entry) |
| orderby | give the order such as mdy, dmy etc where d refers to day, m to month and y to year |

Value

entry corrected entries as in standard date format

Examples

```
convert_date_numeric_stdform(c("01/01/2000", "02/02/2002"), c(1, 2), "dmy")
```

`convert_date_string_stdform`

Helper function to keep date formats in year-month-date

Description

Helper function to keep date formats in year-month-date

Usage

```
convert_date_string_stdform(entry, orderby)
```

Arguments

| | |
|---------|---|
| entry | a date e.g 1 Jan 2020 with no commas |
| orderby | give the order such as mdy, dmy etc where d refers to day, m to month and y to year |

Value

entry corrected entries as in standard date format

Examples

```
convert_date_string_stdform("Jan-1-2020", "mdy")
```

`convert_to_number`

Function that convert a number represented as character array

Description

Function that convert a number represented as character array

Usage

```
convert_to_number(character_array)
```

Arguments

character_array
a character array of numbers

Value

converted_number in numeric form

Examples

```
convert_to_number(c("1", "9", "8"))
```

descriptive_stats_col_excl_nrcode

Function to return descriptive statistics, sum, no of observations, mean, mode. median, range, standard deviation and standard error

Description

Function to return descriptive statistics, sum, no of observations, mean, mode. median, range, standard deviation and standard error

Usage

```
descriptive_stats_col_excl_nrcode(data, column_name, nrcode = NA)
```

Arguments

data data frame
column_name the column name
nrcode non response code corresponding to the column

Value

the descriptive statistics for success , error for failure

Examples

```
descriptive_stats_col_excl_nrcode(data.frame("age" = c(21, 15),  
"Name" = c("John", "Dora")), "age", NA)
```

get_colno_pattern_colname

Function to return the column number if a given pattern is contained in the column names of a data

Description

Function to return the column number if a given pattern is contained in the column names of a data

Usage

```
get_colno_pattern_colname(pattern, column_names)
```

Arguments

pattern a string that needs to be checked
column_names column names actually have

Value

column number, if success error, if failure

Examples

```
get_colno_pattern_colname("age", "female_age")
```

get_columnno_fornames *Function to return the column number for column name*

Description

Function to return the column number for column name

Usage

```
get_columnno_fornames(data, column_name)
```

Arguments

data a data frame
column_name column names of the data frame

Value

column number, if success error, if failure

Examples

```
get_columnno_fornames(data.frame("Age" = c(21, 15),  
"Name" = c("John", "Dora")), "Name")
```

| | |
|-------------------|---|
| get_contents_cols | <i>Function to return the unique contents of the column given the column name</i> |
|-------------------|---|

Description

Function to return the unique contents of the column given the column name

Usage

```
get_contents_cols(data, colname)
```

Arguments

| | |
|---------|---|
| data | a data frame |
| colname | name of column corresponding to year of birth |

Value

the contents of the column, if success error if failure

Examples

```
get_contents_cols(data.frame(  
"yob" = c(1951, 1980),  
"Name" = c("John", "Dora")  
) , "yob")
```

| | |
|-----------------|--|
| get_effect_size | <i>Function to get the effect size</i> |
|-----------------|--|

Description

Function to get the effect size

Usage

```
get_effect_size(data, variable, by, ...)
```

Arguments

| | |
|----------|---|
| data | a data frame |
| variable | variables to be selected for summary |
| by | A column name (quoted or unquoted) in data. |
| ... | extra parameters required |

Value

returns the effect sizes

```
get_mode_from_vector #####
                        Function to return mode
```

Description

Function to return mode

Usage

get_mode_from_vector(v)

Arguments

| | |
|---|----------|
| v | a vector |
|---|----------|

Value

mode

Examples

get_mode_from_vector(c(1, 1, 2, 3))

```
get_sem #####
          Function to estimate standard error of the mean
```

Description

```
##### Func-
tion to estimate standard error of the mean
```

Usage

```
get_sem(x)
```

Arguments

```
x,          a vector
```

Value

SE the standard error of the mean

Examples

```
get_sem(c(1, 2, 3, 4))
```

```
get_summary_gtsummary  Function to return the summary table using gtsummary package
```

Description

Function to return the summary table using gtsummary package

Usage

```
get_summary_gtsummary(the_data, selectvar, byvar = NULL, label = NULL)
```

Arguments

```
the_data    a data frame
selectvar   variables to be selected for summary
byvar       A column name (quoted or unquoted) in data. Summary statistics will be calcu-
            lated separately for each level of the by variable. If NULL, summary statistics
            are calculated using all observations.
label       List of formulas specifying variables labels,
```

Value

the summary using gtsummary's tbl_summary option

Examples

```
trial <- gtsummary::trial
table1 <- get_summary_gtsummary(trial, c("trt", "age", "grade"),
  byvar = "trt")
```

get_value_from_codes *Function to get the actual value of column content if its coded*

Description

Function to get the actual value of column content if its coded

Usage

```
get_value_from_codes(data, column, nrcode = NA, list_codes_values)
```

Arguments

| | |
|-------------------|--|
| data | a data frame |
| column | column name for value |
| nrcode | non response code corresponding to gender column |
| list_codes_values | list of codes to understand the codes and value |

Value

0, if success error if failure

Examples

```
data = data.frame("sex" = c(1, 2, 2, 1, 1),
  "Name" = c("John", "Dora", "Dora", "John", "John"))
list_codes_values = list(c("F", "M"), c(1, 2))
ans <- get_value_from_codes(data, column = "sex", nrcode = NA,
  list_codes_values)
```

```
keep_required_columns #####
                        Function to keep only certain variables
```

Description

```
##### Func-
tion to keep only certain variables
```

Usage

```
keep_required_columns(variables, the_data)
```

Arguments

```
variables    list of variables
the_data     data to be sub setting
```

Value

```
subset
```

Examples

```
the_data <- data.frame("Age" = c(21, 15), "sex" = c("m", "f"))
variable <- "Age"
keep_required_columns(variable, the_data)
```

```
present_mean_sd_rmna_text
                        #####
                        Function to present the mean and sd of a data set in the form Mean
                        (SD)
```

Description

```
##### Func-
tion to present the mean and sd of a data set in the form Mean (SD)
```

Usage

```
present_mean_sd_rmna_text(data, column_name, nrcode = NA)
```

Arguments

| | |
|-------------|---|
| data | data frame |
| column_name | the column name |
| nrcode | non response code corresponding to the column |

Value

the mean(sd), error for failure

Examples

```
present_mean_sd_rmna_text(data.frame(  
  "age" = c(21, 15),  
  "Name" = c("John", "Dora")  
) , "age", NA)
```

represent_categorical_data_exclude_missing

Function to find the number and percentages of categories

Description

Function to find the number and percentages of categories

Usage

```
represent_categorical_data_exclude_missing(data, variable, nrcode = NA)
```

Arguments

| | |
|----------|-------------------|
| data, | a data frame |
| variable | the column name |
| nrcode | non response code |

Value

number and percentages or error if failure

Examples

```
this.df <- data.frame(c(11, 78), c("m", "f"), stringsAsFactors = FALSE)  
colnames(this.df) <- c("mark", "gender")  
represent_categorical_data_exclude_missing(this.df, "gender", NA)
```

```
represent_categorical_data_forsubgroups
```

Function to find the number and percentages of categories

Description

Function to find the number and percentages of categories

Usage

```
represent_categorical_data_forsubgroups(  
  data,  
  variable1,  
  variable2,  
  nrcode = NA  
)
```

Arguments

| | |
|-----------|--|
| data, | a data frame |
| variable1 | the column name of the variable to be grouped based on |
| variable2 | the column name of the variable to be represented |
| nrcode | non response code for the variable2 |

Value

the subgroup

Examples

```
this.df <- data.frame(c(11, 78,22), c("m", "f", "f"), c(1,2,2),  
  stringsAsFactors = FALSE)  
colnames(this.df) <- c("mark", "gender", "group")  
represent_categorical_data_forsubgroups(this.df, "group", "gender", NA)
```

```
represent_categorical_data_include_missing
```

Function to find the number and percentages of categories

Description

Function to find the number and percentages of categories

Usage

```
represent_categorical_data_include_missing(data, variable, nrcode = NA)
```


Arguments

| | |
|----------|-------------------|
| data, | a data frame |
| variable | the column name |
| nrcode | non response code |

Value

number and percentages or error if failure

Examples

```
this.df <- data.frame(c(11, 78), c("m", "f"), stringsAsFactors = FALSE)
colnames(this.df) <- c("mark", "gender")
represent_categorical_data_include_missing(this.df, "gender", NA)
```

represent_categorical_textdata

Function to represent categorical data in the form - numbers (percentage)

Description

Function to represent categorical data in the form - numbers (percentage)

Usage

```
represent_categorical_textdata(data, variable, nrcode)
```

Arguments

| | |
|----------|-------------------|
| data | data frame |
| variable | column name |
| nrcode | non response code |

Value

the numbers (percentage) , error for failure

Examples

```
df <- data.frame(c(11, 78), c("m", "f"), stringsAsFactors = FALSE)
colnames(df) <- c("mark", "gender")
represent_categorical_textdata(df, "gender", NA)
```

represent_numerical_data_forsubgroups

Function to find the number and percentages of categories

Description

Function to find the number and percentages of categories

Usage

```
represent_numerical_data_forsubgroups(data, variable1, variable2, nrcode = NA)
```

Arguments

| | |
|-----------|---|
| data, | a data frame |
| variable1 | the column name of the variable to be grouped based on (categorical column) |
| variable2 | the column name of the variable to be represented (numerical data) |
| nrcode | non response code for the variable2 |

Value

the subgroup

Examples

```
this.df <- data.frame(c(11, 78,22), c("m", "f", "f"), c(1,2,2),  
stringsAsFactors = FALSE)  
colnames(this.df) <- c("mark", "gender", "group")  
represent_numerical_data_forsubgroups(this.df, "group", "mark", NA)
```

return_longitudinal_summary

Function to get the longitudinal summary mean and sd

Description

Function to get the longitudinal summary mean and sd

Usage

```
return_longitudinal_summary(thedata, columnnames, nrcode = NA)
```

Arguments

| | |
|-------------|---|
| thedata | a data frame |
| columnnames | column names of the data that are some observations at some time points |
| nrcode | the non response code in the data |

Value

returns the effect sizes

Examples

```
test_data <- as.data.frame(cbind(c(1,2,3,4,5), c(20,40,60,80,100),
c("F", "F", "M", "M", "F")))
colnames(test_data) <- c("no", "marks", "gender")
test_data$marks <- as.numeric(test_data$marks)
results <- return_longitudinal_summary(test_data, "marks", NA)
```

return_subgroup_omitna

Function to return a subgroup when certain variable equals the given value while omitting those with NA

Description

Function to return a subgroup when certain variable equals the given value while omitting those with NA

Function to return a subgroup when certain variable equals the given value while omitting those with NA

Usage

```
return_subgroup_omitna(data, variable, value)
```

```
return_subgroup_omitna(data, variable, value)
```

Arguments

| | |
|----------|---|
| data | data frame |
| variable | that corresponds to a column |
| value | a value that can be taken by the variable |

Value

subgroup a data frame if success error if failure

subgroup a data frame if success error if failure

Examples

```
return_subgroup_omitna(data.frame(
  "age" = c(21, 15),
  "Name" = c("John", "Dora")
), "age", 10)
return_subgroup_omitna(data.frame(
  "age" = c(21, 15),
  "Name" = c("John", "Dora")
), "age", 10)
```

return_subgroup_withNA

Function to return a subgroup when certain variable equals the given value while omitting those with NA

Description

Function to return a subgroup when certain variable equals the given value while omitting those with NA

Usage

```
return_subgroup_withNA(data, variable, value)
```

Arguments

| | |
|----------|---|
| data | data frame |
| variable | that corresponds to a column |
| value | a value that can be taken by the variable |

Value

subgroup a data frame if success error if failure

Examples

```
return_subgroup_withNA(data.frame(
  "age" = c(21, 15),
  "Name" = c("John", "Dora")
), "age", 10)
```

| | |
|----------|--|
| test_age | <i>Function to check the format of 'age' in data</i> |
|----------|--|

Description

Function to check the format of 'age' in data

Usage

```
test_age(data, agecolumn = "age", nrcode = NA)
```

Arguments

| | |
|-----------|--|
| data | a data frame |
| agecolumn | column name that corresponds to age or date of birth |
| nrcode | non response code corresponding to age column |

Value

0, if success error if failure

Examples

```
df <- data.frame("Age" = c(21, 15), "Name" = c("John", "Dora"))
test_age(df, "age", 999)
```

| | |
|------------------|--|
| test_columnnames | <i>Function to test column names of a data being different from what specified</i> |
|------------------|--|

Description

Function to test column names of a data being different from what specified

Usage

```
test_columnnames(column_names, data)
```

Arguments

| | |
|--------------|--------------------------------|
| column_names | column names of the data frame |
| data | a data frame |

Value

0, if success error, if failure

Examples

```
test_columnnames(c("name", "age"), data.frame(  
  "Age" = c(21, 15),  
  "Name" = c("John", "Dora")  
))
```

test_column_contents *Function to check the format of column contents*

Description

Function to check the format of column contents

Usage

```
test_column_contents(data, column, code, nrcode = NA)
```

Arguments

| | |
|--------|--|
| data | a data frame |
| column | column name for gender |
| code | how column values are coded |
| nrcode | non response code corresponding to gender column |

Value

0, if success error if failure

Examples

```
test_column_contents(data.frame(  
  "sex" = c("m", "f"),  
  "Name" = c("John", "Dora")  
) , "sex", c("m", "f"), 999)
```

test_data_numeric *Function to check the format of a numeric column*

Description

Function to check the format of a numeric column

Usage

```
test_data_numeric(column_name, data, nrcode = NA, minval, maxval)
```

Arguments

| | |
|-------------|---|
| column_name | the column name |
| data | data frame |
| nrcode | non response code corresponding to the column |
| minval | minimum value allowed |
| maxval | maximum value allowed |

Value

0, if success error, if failure

Examples

```
test_data_numeric("age", data.frame(
  "Age" = c(21, 15),
  "Name" = c("John", "Dora")
), -99, 0, 100)
```

test_data_numeric_norange *Function to check the format of a numeric column when the values are not bounded*

Description

Function to check the format of a numeric column when the values are not bounded

Usage

```
test_data_numeric_norange(column_name, data, nrcode = NA)
```

Arguments

| | |
|-------------|---|
| column_name | the column name |
| data | data frame |
| nrcode | non response code corresponding to the column |

Value

0, if success error, if failure

Examples

```
test_data_numeric_norange("marks", data.frame(
  "marks" = c(210, 99),
  "Name" = c("John", "Dora")
), -99)
```

| | |
|------------------|--|
| test_data_string | <i>Function to check the format of a string column</i> |
|------------------|--|

Description

Function to check the format of a string column

Usage

```
test_data_string(data, column_name, nrcode = NA)
```

Arguments

| | |
|-------------|---|
| data | data frame |
| column_name | the column name |
| nrcode | non response code corresponding to the column |

Value

0, if success error, if failure

Examples

```
test_data_string(data.frame("Age" = c(21, 15), "Name" = c("John", "Dora")),
  "name", -999)
```

`test_data_string_restriction`

Function to check the format of a string column when the string values are given

Description

Function to check the format of a string column when the string values are given

Usage

```
test_data_string_restriction(data, column_name, nrcode = NA, allowed_strings)
```

Arguments

| | |
|------------------------------|---|
| <code>data</code> | data frame |
| <code>column_name</code> | the column name |
| <code>nrcode</code> | non response code corresponding to the column |
| <code>allowed_strings</code> | allowed strings or characters to represent meaningful entry |

Value

0, if success error, if failure

Examples

```
test_data_string_restriction(  
  data.frame("Age" = c(21, 15), "sex" = c("m", "f")),  
  "sex", -999, c("f", "m")  
)
```

`test_file_exist_read` *Function to throw error on invalid directory or file and if not readable*

Description

Function to throw error on invalid directory or file and if not readable

Usage

```
test_file_exist_read(filename)
```

Arguments

| | |
|-----------------------|-----------------------|
| <code>filename</code> | name of a file or dir |
|-----------------------|-----------------------|

Value

0, if success error, if failure

Examples

```
test_file_exist_read(system.file("extdata", "blank.txt",  
  package = "IPDFileCheck"  
)  
)
```

| | |
|-------------|--|
| test_gender | <i>Function to check the format of 'gender' column in data</i> |
|-------------|--|

Description

Function to check the format of 'gender' column in data

Usage

```
test_gender(data, gendercode, gendercolumn = "gender", nrcode = NA)
```

Arguments

| | |
|--------------|--|
| data | a data frame |
| gendercode | how gender is coded |
| gendercolumn | column name for gender |
| nrcode | non response code corresponding to gender column |

Value

0, if success error if failure

Examples

```
test_gender(data.frame("sex" = c("m", "f"), "Name" = c("John", "Dora")),  
  c("f", "m"), "sex", 999)
```

Index

calculate_age_from_dob, 2
calculate_age_from_year, 3
check_col_pattern_colname, 5
check_column_exists, 4
check_load_packages, 5
cohensd, 6
convert_date_numeric_stdform, 6
convert_date_string_stdform, 7
convert_to_number, 7

descriptive_stats_col_excl_nrcode, 8

get_colno_pattern_colname, 9
get_columnno_fornames, 9
get_contents_cols, 10
get_effect_size, 10
get_mode_from_vector, 11
get_sem, 12
get_summary_gtsummary, 12
get_value_from_codes, 13

keep_required_columns, 14

present_mean_sd_rmna_text, 14

represent_categorical_data_exclude_missing, 15
represent_categorical_data_forsubgroups, 16
represent_categorical_data_include_missing, 16
represent_categorical_textdata, 17
represent_numerical_data_forsubgroups, 18
return_longitudinal_summary, 18
return_subgroup_omitna, 19
return_subgroup_withNA, 20

test_age, 21
test_column_contents, 22
test_columnnames, 21
test_data_numeric, 23
test_data_numeric_norange, 23
test_data_string, 24
test_data_string_restriction, 25
test_file_exist_read, 25
test_gender, 26