Package 'fdq'

December 11, 2016

Type Package

Title Forest Data Quality

Date 2016-12-11

Version 0.2

Maintainer Caíque de Oliveira de Souza <forestdataqualitysoftware@gmail.com>

Description
Contains methods of analysis of forest databases, the purpose of the analyzes is to evaluate the quality of the data present in the databases focusing on the dimensions of consistency, punctuality
and completeness. Databases can range from forest inventory data to growth model data. The package has
methods to work with large volumes of data quickly, in addition in certain analyzes it is possible to generate the graphs for a better understanding of the analysis and reporting of the analyzed analysis.

License GPL-3

Encoding UTF-8

LazyData true

Depends R(>= 3.0), Fgmutils,data.table

Imports ggplot2, sqldf, randomcoloR, plyr, utils, stats

RoxygenNote 5.0.1

NeedsCompilation no

Author Caíque de Oliveira de Souza [aut, cre],
Clayton Vieira Fraga Filho [ctb, dtc],
Miquéias Fernandes [ctb]

Repository CRAN

Date/Publication 2016-12-11 20:35:12

R topics documented:

`calculate_jumps_ages` *calculate_jumps_ages*

Description

This function generates the jumps per diameter class among the ages in the database

Usage

```
calculate_jumps_ages(database, id_field, rounded_ages_field,
  diameter_class_field)
```

Arguments

database	data.frame data.table
id_field	string name of column with unique identifier of each tree
rounded_ages_field	string name of column with rounded ages
diameter_class_field	string name of column with diametric classes

---

check_ages *check_ages*

Description

This analysis verifies age differences on a paired basis, if the rounded ages are in months the check is if the difference is 12 months, if it is in year the consecutive ages should only present difference of 1 year, doubts about how to pair your base consult The Fgmutils package

Usage

```
check_ages(data_base, rounded_age1, rounded_age2, months = FALSE)
```

Arguments

data_base	data.frame data.table
rounded_age1	string name of column rounde age one
rounded_age2	string name of column rounde age two
months	TRUE for age in months or FALSE for age in years

---

check_clones_different_plot *check_clones_different_plot*

Description

This function checks if the clones of a tree have different plots

Usage

```
check_clones_different_plot(database, parcel_name, clone_name,
  variables_to_group)
```

Arguments

database	data.frame, data.table or any database
parcel_name	string name of the field containing the parcels
clone_name	string name of the field containing the clones
variables_to_group	string(s) variable (s) that you want to group the result of the analysis

---

check_dead_state *check_dead_state*

Description

This function checks if the base state field is equal to dead (M) and there is some kind of measurement

Usage

```
check_dead_state(data_base, state, measurement_variables)
```

Arguments

data_base	data.frame data.table or any database
state	string field name representing state column in database
measurement_variables	string vector that contains a set of measurement variables to be analyzed, this variables are names of columns in database

---

check_diametric_increase *check_diametric_increase*

Description

This function is an implementation of the process proposed by Demolinari (2006) in his master's thesis that is the diametric increase, given a plot it is possible to follow all the growth of the trees contained in the same at all ages present in the plot, this is done through Of colored color graphics where each color represents a class of diameter and a meditation that as an advance for each age is generated a new graphic that contains as changes represented by cores or a quantity of trees that migrate of class, this can have an identification Of inconsistent behaviors within the plot.

Usage

```
check_diametric_increase(database, id_field, plot_field, plot_analyzed,
  ages_field, rounded_ages_field, diameter_classes_field, database_nhaclasses,
  plot_field_database_nhaclassses, diameter_classes_field_database_nhaclasses,
  rounded_ages_field_database_nhaclasses, nha_classes_database_nhaclasses)
```

Arguments

database,	data.table, data.frame or any database
id_field,	string with the name of column containing the unique identifier of each tree
plot_field,	string with the name of column containing the plots of database
plot_analyzed,	the number of plot to be analyzed, example: 356
ages_field,	string with the name of column containing the ages of each tree
rounded_ages_field,	string with the name of column containing the rounded ages of each tree, example: 12, 36
diameter_classes_field,	string with the name of column containing the diameter class of each tree
database_nhaclasses,	data.table, data.frame, or any database containg the NHa Class of database used in this analysis. To obtain this database you simply have to submit the original base the generate_number_hectare_class function contained in this package
plot_field_database_nhaclassses,	string with the name of column containing the plots of database resulting from the generate_number_hectare_class function
diameter_classes_field_database_nhaclasses,	string with the name of column containing the diameter class of each tree of database resulting from the generate_number_hectare_class function
rounded_ages_field_database_nhaclasses,	string with the name of column containing the rounded ages of each tree of database resulting from the generate_number_hectare_class function
nha_classes_database_nhaclasses,	string with the name of column containing the NHa Class of database resulting from the generate_number_hectare_class function

---

check_existing_ages *check_existing_ages*

Description

This function checks if a given set of ages exists in a database column

Usage

```
check_existing_ages(database, ages_name, ages_to_check)
```

Arguments

database	data.frame data.table or any database
ages_name	string name of the column representing ages
ages_to_check	string name/vector of the column (s) representing ages to be checked

---

check_existing_place *check_existing_place*

Description

This function checks whether a particular set of sites or locations exists in a database column

Usage

```
check_existing_place(database, place_name, places_to_check)
```

Arguments

database	data.frame, data.table or any database
place_name	string name of the column representing site or place
places_to_check	value(s) to be checked, example: c(12,21,33)

---

check_existing_plots *check_existing_plots*

Description

This function checks if a particular set of parcels exists in a database column

Usage

```
check_existing_plots(database, plots_name, plots_to_check)
```

Arguments

database	data.frame, data.table or any database
plots_name	string column name representing parcels in the base
plots_to_check	value(s) to be checked, example: c(356,122)

check_measurements_state *check_measurements_state*

Description

This function checks if there is a measurement variable with value equal to 0 and if the respective states are different from M, F, A

Usage

```
check_measurements_state(data_base, measurement_variables, state)
```

Arguments

data_base	data.frame, data.table or any database
measurement_variables	set of variables to be analyzed, this set can be a vector of string with names of colunms
state	string name of the field that represents the state in database

check_measurement_ages *check_measurement_ages*

Description

This function verifies if measurement variables have records of type DAP2 <DAP1, HT2 <HT1 in consecutive ages i + 1 and i tt is necessary that the base is already paired to perform such analysis, to know more about pairing consult the Fgmutils package

Usage

```
check_measurement_ages(data_base, measurement_variable1, measurement_variable2)
```

Arguments

data_base	data.frame, data.table or any database
measurement_variable1	string field containing the measurement variables at age 1
measurement_variable2	string field containing the measurement variables at age 2

`check_parcel_different_spacing` *check_parcel_different_spacing*

Description

This function checks for partitions with different spacing at i and i + 1 ages, it is necessary that the base be paired including the field representing the spacing, doubts about how to pair its base see the Fgmutils package

Usage

```
check_parcel_different_spacing(database, parcel_name, spacing_age1,
  spacing_age2, variables_to_group)
```

Arguments

`database`	data.frame, data.table or any database
`parcel_name`	string containing the field name parcels in database
`spacing_age1`	string containing the name of the field spacing in the first age
`spacing_age2`	string containing the name of the field spacing in the second age
`variables_to_group`	variable (s) that you want to group the result of the analysis, this can be a vector os strings or strign name to group

`check_size_age_plot` *check_size_age_plot*

Description

This function checks if the age field is more than one age, returning TRUE to for yes and FALSE for no

Usage

```
check_size_age_plot(database, age_name)
```

Arguments

`database`	data.frame, data.table or any database
`age_name`	string containing the name of the column that represents age

check_undefined_spacing *check_undefined_spacing*

Description

This function checks if there is any record with undefined spacing (0 or NA)

Usage

```
check_undefined_spacing(data_base, spacings)
```

Arguments

data_base	data.frame, data.table or any database
spacings	string vector containing the name of the variable (s) than represent spacings in database

check_variables *check_variables*

Description

This function checks if the entered column exists within the base

Usage

```
check_variables(database, variables)
```

Arguments

database	data.frame, data.table or any database
variables	vector of strings with names of columns

Value

TRUE for all variables in database, or FALSE for variables not present in columns

Examples

```
test <- data.frame("tree","diametrer","N")
check_variables(test,c("tree","diameter"))
```

check_zero_measurement *check_zero_measurement*

Description

This analysis verifies which measurement variables have values equal to 0 and then checks if there are variables in the states that the user reported

Usage

```
check_zero_measurement(data_base, measurement_variables, state_name,
  states_to_check)
```

Arguments

data_base	data.frame, data.table or any database
measurement_variables	string vector containing name of the field(s) it represents measurement variable(s) to be analyzed
state_name	string vector containing the name of the variable than represents state in database
states_to_check	string vector containing the name of the the states to be checked, the user can inform this names in a string vector like ("F","N")

correct_probability *correct_probability*

Description

This function will adjust the probability after generation of surviving tree classes per hectare

Usage

```
correct_probability(database, plot_field, rounded_age_field, probability_field,
  nha_class_field)
```

Arguments

database	data.frame, data.table or any database
plot_field	string name of column with plots of database
rounded_age_field	string name of column with rounded ages, examples: 12,24,36,48

probability_field
: string name of column with probabilitys

nha_class_field
: string name of column with NHa (surviving tree classes per hectare) classes,

---

`find_missing_age` *find_missing_age*

Description

This function identifies the missing age values in the database and notifies them to the user.

Usage

```
find_missing_age(database, age_name, ages_to_check)
```

Arguments

database	data.frame, data.table or any database
age_name	string that contains the field name that represents age in database
ages_to_check	vector containing the values of ages to be checked like c(12,23,48)

`find_missing_place` *find_missing_place*

Description

This function identifies values of sites or locations in the database and notifies them to the user

Usage

```
find_missing_place(database, place_name, places_to_check)
```

Arguments

database	data.frame, data.table or any database
place_name	string that contains the field name representing site or place in database
places_to_check	vector containing the values of places/sites to be checked like c(21,33,48)

`find_missing_variable` *find_missing_variable*

Description

This function identifies non-existent column names in the database and informs the user

Usage

```
find_missing_variable(data_base, variables)
```

Arguments

`data_base`	data.frame, data.table or any database
`variables`	vector string that contains the name(s) of columns to be checked in database

`generate_colors_diameter_class` *generate_colors_diameter_class*

Description

This function generates a new random color for each diameter class in the base

Usage

```
generate_colors_diameter_class(database, diameter_classe_name)
```

Arguments

`database`	data.frame, data.table or any database
`diameter_classe_name`	string with the name of field (column) containing the diameter classes

generate_diameter_classes *generate_diameter_classes*

Description

This function identifies non-existent column names in the database and informs the user

Usage

```
generate_diameter_classes(database, diameter_names, amplitude,
  name_of_diameter_class)
```

Arguments

database	data.frame, data.table or any database
diameter_names	string with name of the field that contains the diameters of database
amplitude	desired amplitude for class creation, example: 1,2,4,6,7
name_of_diameter_class	string with name you want for the field class of diameter

generate_initial_diameter_class *generate_initial_diameter_class*

Description

This function generates the initial class field to aid in the process of diametric increasing

Usage

```
generate_initial_diameter_class(database, plot_name, age_name)
```

Arguments

database	data.frame, data.table or any database
plot_name	string with the name of field representing plots in database
age_name	string with the name of field representing rounded age

generate_limiar *generate_limiar*

Description

This function generates a threshold for certain values based on the field entered and a maximum value for setting

Usage

```
generate_limiar(database, field, max)
```

Arguments

database,	data.table, data.frame or any database
field,	string name with column you want to work on
max,	number with the maximum value you want to establish the threshold

generate_new_color *generate_new_color*

Description

This function generates a new random color without repeating the ones that were entered in the last field as parameter

Usage

```
generate_new_color(colors)
```

Arguments

colors	vector of strings containing existing colors, exemple: c("#6140bc" "#e75bf7" "#d15102" "#6a0b9e" "#e8ad4e")

generate_number_hectare *generate_number_hectare*

Description

This function generates the NHa, field that represents the number of surviving trees per hectare

Usage

```
generate_number_hectare(database, area_name, n_name, nha_name = "NHa")
```

Arguments

database	data.frame, data.table or any database
area_name	string with the name of field containing area in database
n_name	string with the name of field containing numbers of trees in database
nha_name	string with name you want for the field number of trees per hectare

generate_number_hectare_class *generate_number_hectare_class*

Description

This function generates the NHa classes column which represents the classes with the number of surviving trees per hectare

Usage

```
generate_number_hectare_class(database, nha_field, n_field, area_field,
  plot_field, rounded_age_field, age_field, diameter_class_field, state_field,
  dap_field, amplitude = 2)
```

Arguments

database	data.frame, data.table or any database
nha_field,	name of the column containing the number of trees surviving per hectare
n_field,	name of the column containing the number of trees in a plot, or diameter class
area_field,	name of the column containing the area
plot_field,	name of the column containing the plots
rounded_age_field,	name of the column containing the rounded ages, example: 12,24,36

age_field,	name of the column containing the ages
diameter_class_field,	name of the column containing the diameter classes
state_field,	name of the column containing the states of trees
dap_field,	name of the column containing the diameter (DAP) of trees
amplitude,	integer integer number with the amplitude of diameter classes, example: 1, 2, 3

---

get_ages *get_ages*

Description

This function concatenates age values in a string for a query and returns the same

Usage

```
get_ages(database, age_name, age_values)
```

Arguments

database	data.frame, data.table or any database
age_name	string with the name of field (column) containing the ages
age_values	vector with the age values you want to assemble string to made query, example: c(12,24,36)

---

get_max *get_max*

Description

This function returns the maximum value of one or more fields of measurement variables

Usage

```
get_max(database, variables)
```

Arguments

database	data.frame, data.table or any database
variables	string vector with name(s) of the column (s) you want to know the maximum value

get_min *get_min*

Description

This function returns the minimum value of one or more fields of measurement variables

Usage

```
get_min(database, variables)
```

Arguments

database	data.frame, data.table or any database
variables	string vector with name(s) of the column (s) you want to know the minimum value

get_place *get_place*

Description

This function returns a database from a particular site or location present in the original database

Usage

```
get_place(database, place_name, place_value)
```

Arguments

database	data.frame, data.table or any database
place_name	string with the name of the column that represents the place
place_value	vector with values of that you want to filter the sites/places of the database

mount_list *mount_list*

Description

This function serves as a help for the check_diametric_increase function, it generates lists of values to be plotted as columns

Usage

```
mount_list(database, fieldColorName = "color", fieldQuantityName = "n",
  diameter_classes_name = "classedediametro")
```

Arguments

database,	data.table, data.frame or any database
fieldColorName,	string with the name you want for the color column
fieldQuantityName,	string name of column with number of trees
diameter_classes_name,	string name of column with the diameter classes

mount_query *mount_query*

Description

This auxiliary function checks that need to group fields of certain measurements

Usage

```
mount_query(database, select_names, group_names, option)
```

Arguments

database	data.frame, data.table or any database
select_names	string vector with the name(s) of the column(s) you want to include in the selection
group_names	string vector with the name(s) of the column(s) you want to group the results
option	options to make the query, can be 1,2,3 each one for one use in the analysis functions

plot_cols *plot_cols*

Description

This function serves as a help for the check_diametric_increase function, it generates a chart based on the lists created by the mount_list function

Usage

```
plot_cols(list, name, my_title, ceil = NaN)
```

Arguments

list,	list of columns generated by function mount_list with values and colors to be ploted
name,	string name with the name you want for the file png chart
my_title,	string with the name you want for the title of chart
ceil,	the maximum value to axis, this value will be applied in NHa or N

sort_columns_crescent *sort_columns_crescent*

Description

Sorts the database incrementally based on the selected column

Usage

```
sort_columns_crescent(database, column)
```

Arguments

database	data.frame, data.table or any database
column	string with the name of the column you want sort the database

Index