Package 'dataframeexplorer'

October 13, 2022

Title Familiarity with Dataframes Before Data Manipulation

Version 1.0.2

Description Real life data is muddy, fuzzy and unpredictable. This makes data manipulations tedious and bringing the data to right shape alone is a major chunk of work. Functions in this package help us get an understanding of dataframes to dramatically reduces data coding time.

Depends R (>= 3.3.0)

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.1

Imports tibble, data.table, magrittr, openxlsx, dplyr, plyr, tidyr, stringr

NeedsCompilation no

Author Ashrith Reddy [aut, cre]

Maintainer Ashrith Reddy <ashrithssreddy@gmail.com>

Repository CRAN

Date/Publication 2022-04-04 07:10:02 UTC

R topics documented:

detect_const_cols *Detect if any column of a data.frame has constant values.*

Description

It occasionally happens that a column in dataframe contains a single value throughout. This could lead to redundant computational cost and unexpected behavior in Machine Learning methods. This function scans though all columns of dataframe to examine if any column has no variation.

Usage

```
detect_const_cols(dataset, return_type = "col_names", ignore_na = F)
```

Arguments

dataset	A data.frame
return_type	How to return detected constant columns Use "col_names", "col_positions" or "dataset" to return dataset with deleted constant columns
ignore_na	Whether NA should be ignored while checking if a column has just 1 unique value

Value

. A vector of constant column-names or column positions or dataset with deleted constant columns. Use return_type parameter to specify.

Examples

```
## Not run:
detect_const_cols(dataset = head(mutate(mtcars, mpg_2 =  999)))

## End(Not run)
```

detect_dupl_cols *Detect if any column of a data.frame is a duplicate of another*

Description

It occasionally happens that 2 (or more) columns in dataframe are exactly identical. This could lead to redundant computational cost and unexpected behavior in Machine Learning methods. This function scans though all column combinations of dataframe to examine if any 2 columns are exactly identical.

Usage

```
detect_dupl_cols(dataset, return_type = "col_names", duplicate_col = "right")
```

Arguments

dataset	A data.frame
return_type	How to return detected duplicate columns Use "col_names", "col_positions" or "dataset" to return dataset with deleted duplicate columns
duplicate_col	If 2 columns are identical, which of the 2 columns should be treated as duplicate? Use "right" for right column, "left" for left.

Value

A vector of duplicate column names or column positions or dataset with deleted duplicate columns. Use return_type parameter to specify.

Examples

```
## Not run:
detect_dupl_cols(dataset = head(mutate(mtcars, mpg_2 =  mpg)), duplicate_col = "right")

## End(Not run)
```

---

frequency_table *Generate frequency of each entry in each column of dataframe*

Description

Real-life data is seldom perfect and fields in a data.frame contains entries not anticipated by the data scientist. This function helps to know your data entries before performing any manipulations on it. This function generates frequency table excel, each column of input dataframe in a separate sheet in output excel file. Warning: An excel sheet can support 2^20 rows of data only (approx. 1 million). If the number of unique entries in a column exceeds that, excel will drop the low frequency entries.

Usage

```
frequency_table(
  dataset,
  output_filename = "",
  maximum_entries = 2^20,
  format_width = TRUE,
  sl_no_required = TRUE,
  frequency_required = TRUE,
  percentage_required = TRUE,
  cumulative_percentage_required = FALSE,
  string_length_required = TRUE
)
```

Arguments

dataset	A data.frame
output_filename	Name of the output text file (should end in ".xlsx") Strongly advised to pass this parameter, else the function's default is "frequency_table_<system_time>.xlsx"
maximum_entries	Maximum unique entries in output. For e.g. setting this parameter to 10000 will return only top 10000 occurring entries in each column
format_width	Boolean input indicating if output excel cells' column width need to be formatted to "auto"
sl_no_required	Boolean input indicating if Sl_No column needs to be present in output excel
frequency_required	Boolean input indicating if Frequency column needs to be present in output excel
percentage_required	Boolean input indicating if Percentage column needs to be present in output excel
cumulative_percentage_required	Boolean input indicating if Cumulative_Percentage column needs to be present in output excel
string_length_required	Boolean input indicating if String_Length column needs to be present in output excel

Value

Does not return to calling function, writes to file system rather

Examples

```
## Not run:
frequency_table(dataset = iris, output_filename = "frequency_table_iris.xlsx")
frequency_table(dataset = mtcars, output_filename = "C/Users/Desktop/frequency_table_mtcars.xlsx")

## End(Not run)
```

---

glimpse_to_file *Generate glimpse of dataset*

Description

Understanding the dataset through a glimpse of it will come handy while data manipulation coding. This function generates the glimpse of data.frame (similar to str()) using tibble::glimpse and write to a text file. Using same file name for different datasets will append the outputs to a same file.

Usage

```
glimpse_to_file(dataset, output_filename = "")
```

Arguments

dataset	A data.frame object
output_filename	Name of the output text file (prefer to end in ".txt", although the backend will append if not) Function's default is "glimpse_<system_time>.txt"

Value

Does not return any value, writes to disk rather

Examples

```
## Not run:
glimpse_to_file(dataset = mtcars, output_filename = "glimpse_mtcars.txt")
glimpse_to_file(dataset = iris, output_filename = "C/Users/Desktop/glimpse_iris.txt")

## End(Not run)
```

---

level_of_data *Determine the level / primary key of dataset*

Description

Knowing the level of dataset is paramount to effectively and efficiently manipulate data, and the level of dataset is unknown oftentimes. This function checks for count of unique records in all possible column combinations to determine the level of dataset. Check for text file generated for column combinations with unique records.

Usage

```
level_of_data(dataset, output_filename = "", verbose = TRUE)
```

Arguments

dataset	A data.frame
output_filename	Name of the output text file (should end in ".txt", although the backend will append if not) Function's default is "level_of_dataset_<system_time>.txt"
verbose	Pass TRUE for detailed output

Value

Does not return to calling function, writes to file system rather

Examples

```
## Not run:
level_of_data(dataset = iris[,c("mpg", "cyl", "disp", "hp")], output_filename = "level_mtcars.txt")

## End(Not run)
```

`percentiles_table` *Generate percentiles for entire dataframe*

Description

This function generates percentiles for all numeric columns in the dataframe. This will come handy while understanding the distribution of data and in outlier treatment.

Usage

```
percentiles_table(
  dataset,
  output_filename = "",
  percentiles = c(0:10, seq(10, 90, 10), seq(25, 75, 25), 91:100),
  format_width = TRUE,
  sd_required = TRUE,
  min_required = TRUE,
  max_required = TRUE,
  mean_required = TRUE,
  missing_percentage_required = TRUE,
  class_required = TRUE
)
```

Arguments

`dataset`	A data.frame
`output_filename`	Name of the output excel file (should end in ".xlsx") Strongly advised to pass this parameter, else the function's default is "percentiles_table_<system_time>.xlsx"
`percentiles`	numeric vector of probabilities with values in [0,100]
`format_width`	Boolean input indicating if output excel cells' column width need to be formatted to "auto"
`sd_required`	Boolean input indicating if standard deviation column needs to be present in output excel
`min_required`	Boolean input indicating if minimum column needs to be present in output excel
`max_required`	Boolean input indicating if maximum column needs to be present in output excel
`mean_required`	Boolean input indicating if mean column needs to be present in output excel
`missing_percentage_required`	Boolean input indicating if missing percentage column needs to be present in output excel

`class_required` Boolean input indicating if datatype column should be the last column in output excel

Value

Does not return to calling function, writes to file system rather

Examples

```
## Not run:
percentiles_table(mtcars, output_filename = "percentiles_table_mtcars.xlsx")
percentiles_table(iris, output_filename = "C/Users/Desktop/percentiles_table_iris.xlsx")

## End(Not run)
```

Index