Package 'cleanTS'

August 26, 2021

Type Package

Title Testbench for Univariate Time Series Cleaning

Version 0.1.0

Description A reliable and efficient tool for cleaning univariate time series data. It implements reliable and efficient procedures for automating the process of cleaning univariate time series data. The package provides integration with already developed and deployed tools for missing value imputation and outlier detection. It also provides a way of visualizing large time-series data in different resolutions.

License GPL (>= 3)

URL https://github.com/Mayur1009/cleanTS

BugReports https://github.com/Mayur1009/cleanTS/issues

Imports anomalize, data.table, gganimate, ggplot2, ggtext, transformr, glue, imputeTestbench, imputeTS, lubridate, shiny, stringr, tibble, tibbletime

Suggests rmarkdown, gifski (>= 1.4.3), timetk

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author Mayur Shende [aut, cre] (<https://orcid.org/0000-0002-1738-2573>), Neeraj Bokde [aut] (<https://orcid.org/0000-0002-3493-9302>), Andrés E. Feijóo-Lorenzo [aut] (<https://orcid.org/0000-0003-3172-7037>)

Maintainer Mayur Shende <mayur.k.shende@gmail.com>

Repository CRAN

Date/Publication 2021-08-26 19:00:02 UTC

R topics documented:

`animate_interval` *Generate animated plot*

Description

`animate_interval()` creates an animated plot using a `cleanTS` object and a interval.

Usage

```
animate_interval(obj, interval)
```

Arguments

`obj`	A *cleanTS* object.
`interval`	A numeric or character, specifying the viewing interval.

Value

A list containing:

- animation: A `gganim` object.
- nstates: The number of states in the animation.

Examples

```
# Convert sunspots.month to dataframe
data <- timetk::tk_tbl(sunspot.month)

# Randomly insert missing values to simulate missing value imputation
set.seed(10)
ind <- sample(nrow(data), 100)
data$value[ind] <- NA

# Perform cleaning
cts <- cleanTS(data, date_format = "my", time = "index", value = "value")

# Create a `gganim` using `animate_interval()` function
a <- animate_interval(cts, "10 year")
```

---

check_input *Check input data*

Description

This function is used to check and verify the input data given as input. The package needs a univariate time series as input. This function keeps the first 2 columns, first is renamed as time and second is renamed as value. If the optional `time` and `value` arguments are provided then they are used to determine the relevant columns in the data.

Usage

```
check_input(df, dt_format, time, value)
```

Arguments

`df`	A data frame containing the input data. If it contains more than two columns then specify the names of time and value columns using the `time` and `value` arguments.
`dt_format`	Format of timestamps used in the data. It uses lubridate formats as mentioned here.
`time`	The name of column in provided data to be used as time column.
`value`	The name of column in provided data, to be used as value(observations) column.

Value

Data containing 2 columns, time and value. Time column is converted to POSIX object and value to numeric.

`cleanTS` *Clean univariate time-series data*

Description

`cleanTS()`is the main function of the package which creates a cleanTS object. It performs all the different data cleaning tasks, such as converting the timestamps to proper format, imputation of missing values, handling outliers, etc.

Usage

```
cleanTS(
  data,
  date_format,
  imp_methods = c("na_interpolation", "na_locf", "na_ma", "na_kalman"),
  time = NULL,
  value = NULL,
  replace_outliers = TRUE
)
```

Arguments

`data`	A data frame containing the input data. By default, it considers that the first column to contain the timestamps and the second column contains the observations.If that is not the case or if it contains more than two columns then specify the names of time and value columns using the `time` and `value` arguments.
`date_format`	Format of timestamps used in the data. It uses lubridate formats as mentioned here. More than one formats can be using a vectors of strings.
`imp_methods`	The imputation methods to be used.
`time`	Optional, the name of column in provided data to be used as time column.
`value`	Optional, the name of column in provided data, to be used as value column.
`replace_outliers`	Boolean, if `TRUE` then the outliers found will be removed and imputed using the given imputation methods.

Value

A `cleanTS` object which contains:

- Cleaned data
- Missing timestamps
- Duplicate timestamps
- Imputation errors
- Outliers
- Outlier imputation errors

Examples

```
# Convert sunspots.month to dataframe
data <- timetk::tk_tbl(sunspot.month)
print(data)

# Randomly insert missing values to simulate missing value imputation
set.seed(10)
ind <- sample(nrow(data), 100)
data$value[ind] <- NA

# Perform cleaning
cts <- cleanTS(data, date_format = "my", time = "index", value = "value")
print(cts)
```

`detect_outliers` *Find outliers in the data*

Description

This function detects outliers/anomalies in the data. If the `replace_outlier` argument is set to `TRUE`, then the outliers are removed and imputated using the provided imputation methods.

Usage

```
detect_outliers(dt, replace_outlier, imp_methods)
```

Arguments

`dt`	A data.table.
`replace_outlier`	Boolean, defaults to `TRUE`. Specify if the outliers are to be removed and imputed.
`imp_methods`	The imputation methods to be used.

Value

The outliers found in the data. If the outliers are replaced, then the imputation errors are also returned.

duplicate_timestamps *Duplicate Timestamps*

Description

This function finds and removes the duplicate timestamps in the time columns of the data.

Usage

```
duplicate_timestamps(dt)
```

Arguments

dt	Input data

Value

A list of data.table without duplicate timestamps and the duplicate timestamps.

find_dif *Helper function to find the time difference between two given timestamps.*

Description

Helper function to find the time difference between two given timestamps.

Usage

```
find_dif(time1, time2)
```

Arguments

time1	POSIXt or Date object.
time2	POSIXt or Date object.

Value

String, specifying the time interval between `time1` and `time2`. It contains a integer and the unit, for e.g., *5 weeks*, *6 months*, *14 hours*, etc.

gen.animation *Generate animation*

Description

This function takes the list outputted by `animate_interval()` and generates a GIF animation. It is a simple wrapper around the `gganimate::animate()` function with some defaults. The generated GIF can be saved using the `anim_save()` function.

Usage

```
gen.animation(anim, nframes = 2 * anim$nstates, duration = anim$nstate, ...)
```

Arguments

`anim`	List outputted by the `animate_interval()` function containing a `gganim` object and the number of states in the animation.
`nframes`	Number of frames. Defaults to double the number of states in the animation.
`duration`	The duration of animation. Defaults to the number of states in the animation.
`...`	Extra arguments passed to `gganimate::animate()`.

Value

Does not return any value.

Examples

```
## Not run:
a <- animate_interval(cts, "10 year")

# Generate animation using `gen.animation()`
gen.animation(a, height = 700, width = 900)

# Save animation using `anim_save()`
anim_save("filename.gif")

## End(Not run)
```

gen.report *Generate a report.*

Description

gen.report() generates a report of the entire process and the changes made to the original data.

Usage

```
gen.report(obj)
```

Arguments

obj	A *cleanTS* object.

Value

Does not return any value.

Examples

```
# Convert sunspots.month to dataframe
data <- timetk::tk_tbl(sunspot.month)

# Randomly insert missing values to simulate missing value imputation
set.seed(10)
ind <- sample(nrow(data), 100)
data$value[ind] <- NA

# Perform cleaning
cts <- cleanTS(data, date_format = "my", time = "index", value = "value")

gen.report(cts)
```

impute *Handle missing values in the data*

Description

This function handles missing values in the data. It compares various imputation methods and finds the best one for imputation.

Usage

```
impute(dt, methods)
```

Arguments

dt	A data.table.
methods	The imputation methods to be used.

Value

A data.table with missing data imputed, and the imputation errors.

---

interact_plot *Create interactive plot*

Description

Interactive plot is similar to the animated plot, but gives the used some control over the animation. It runs a shinyApp instead of creating a GIF.

Usage

```
interact_plot(obj, interval)
```

Arguments

obj	A *cleanTS* object.
interval	A numeric or character, specifying the viewing interval.

Value

Does not return any value.

Examples

```
## Not run:
# Using the same data used in `cleanTS()` function example.
interact_plot(cts, interval = "1 week")

## End(Not run)
```

mergecsv *Merge Multiple CSV files*

Description

`mergecsv()` takes a folder containing CSV files and merges them into a single *data.table*. It is assumed that the first column of all the CSVs contains the timestamps.

Usage

```
mergecsv(path, formats)
```

Arguments

`path`	Path to the folder.
`formats`	Datetime formats.

Value

Merged `data.table`.

missing_timestamps *Missing timestamps*

Description

This function finds and inserts the missing timestamps in the time columns of the data. The observations for the inserted timestamps are filled with `NA`.

Usage

```
missing_timestamps(dt)
```

Arguments

`dt`	Input data

Value

A list of data.table with inserted missing timestamps and the missing timestamps.

print.cleanTS *Print a* cleanTS *object*

Description

Print method for cleanTS class.

Usage

```
## S3 method for class 'cleanTS'
print(x, ...)
```

Arguments

x	cleanTS object
...	Other arguments

Value

Does not return any value.

Examples

```
## Not run:
# Using the same data as in `cleanTS()` function example.
cts <- cleanTS(data, "my")
print(cts)

## End(Not run)
```

Index