Package ‘timetk’

March 18, 2020

Type Package

Title A Tool Kit for Working with Time Series

Version 0.1.3

Description Get the time series index (date or date-time component),
time series signature (feature extraction of date or date-
time component for time series machine learning),
and time series summary (summary attributes about time series).
Create future time series based on properties of
existing time series index using logistic regression.
Coerce between time-based tibbles (‘tbl’) and ‘xts’, ‘zoo’, and ‘ts’.
Methods discussed herein are commonplace in machine learning, and have been cited in various
literature. Refer to “Calendar Effects” in papers such as

URL https://github.com/business-science/timetk

BugReports https://github.com/business-science/timetk/issues

License GPL (>= 3)

Encoding UTF-8

LazyData true

Depends R (>= 3.3.0), recipes (>= 0.1.4)

Imports dplyr (>= 0.7.0), lazyeval (>= 0.2.0), lubridate (>= 1.6.0),
padr (>= 0.3.0), purrr (>= 0.2.2), readr (>= 1.0.0), stringi
(>= 1.1.5), tibble (>= 1.2), tidyr (>= 0.6.1), xts (>= 0.9-7),
zoo (>= 1.7-14), rlang (>= 0.1.2)

Suggests workflows, parsnip, glmnet, rsample, yardstick, tidyverse,
broom, forcats, forecast, knitr, rmarkdown, robets, scales,
stringr, testthat, tidyquant, fracdiff, timeDate, timeSeries,
tseries, roxygen2

RoxygenNote 7.1.0

VignetteBuilder knitr
**Description**

`step_timeseries_signature` creates a specification of a recipe step that will convert date or date-time data into many features that can aid in machine learning with time-series data.

**Usage**

```r
step_timeseries_signature(
  recipe,
  ..., 
  role = "predictor", 
  trained = FALSE, 
  columns = NULL, 
  skip = FALSE, 
  id = rand_id("timeseries_signature")
)
```
## S3 method for class 'step_timeseries_signature'
tidy(x, ...)

### Arguments

- **recipe**: A recipe object. The step will be added to the sequence of operations for this recipe.
- **...**: One or more selector functions to choose which variables that will be used to create the new variables. The selected variables should have class `Date` or `POSIXct`. See `recipes::selections()` for more details. For the tidy method, these are not currently used.
- **role**: For model terms created by this step, what analysis role should they be assigned? By default, the function assumes that the new variable columns created by the original variables will be used as predictors in a model.
- **trained**: A logical to indicate if the quantities for preprocessing have been estimated.
- **columns**: A character string of variables that will be used as inputs. This field is a placeholder and will be populated once `recipes::prep.recipe()` is used.
- **skip**: A logical. Should the step be skipped when the recipe is baked by `bake.recipe()`? While all operations are baked when `prep.recipe()` is run, some operations may not be able to be conducted on new data (e.g. processing the outcome variable(s)). Care should be taken when using `skip = TRUE` as it may affect the computations for subsequent operations.
- **id**: A character string that is unique to this step to identify it.
- **x**: A `step_timeseries_signature` object.

### Details

- **Date Variable**: Unlike other steps, `step_timeseries_signature` does not remove the original date variables. `recipes::step_rm()` can be used for this purpose.
- **Scaling index.num**: The `index.num` feature created has a large magnitude (number of seconds since 1970-01-01). It’s a good idea to scale and center this feature (e.g. use `recipes::step_normalize()`).

### Removing Unnecessary Features

By default, many features are created automatically. Unnecessary features can be removed using `recipes::step_rm()`.

### Value

For `step_timeseries_signature`, an updated version of `recipe` with the new step added to the sequence of existing steps (if any). For the tidy method, a tibble with columns `terms` (the selectors or variables selected), `value` (the feature names).

### See Also

`recipes::step_holiday()` `recipes::step_date()` `recipes::step_rm()` `recipes::recipe()` `recipes::prep.recipe()` `recipes::bake.recipe()`
Examples

```r
library(recipes)
library(tidyverse)
library(tidyquant)
library(timetk)

FB_tbl <- FANG %>% filter(symbol == "FB")

# Create a recipe object with a timeseries signature step
rec_obj <- recipe(adjusted ~ ., data = FB_tbl) %>%
  step_timeseries_signature(date)

# View the recipe object
rec_obj

# Prepare the recipe object
prep(rec_obj)

# Bake the recipe object - Adds the Time Series Signature
bake(prep(rec_obj), FB_tbl)

# Tidy shows which features have been added during the 1st step
# in this case, step 1 is the step_timeseries_signature step
 tidy(rec_obj)
tidy(rec_obj, number = 1)
```

**Description**

The `timetk` package combines a collection of coercion tools for time series analysis.

**Details**

The `timetk` package has several benefits:

1. Index extraction: get the time series index from any time series object.
2. Understand time series: create a signature and summary from a time series index.
3. Build future time series: create a future time series from an index.
4. Coerce between time-based tibbles (tbl) and the major time series data types xts, zoo, zooreg, and ts: Simplifies coercion and maximizes time-based data retention during coercion to regularized time series (e.g. ts).

To learn more about `timetk`, start with the vignettes: `browseVignettes(package = "timetk")`
Internal Functions Used in timetk

Description
The following are internal functions that are not meant to be used by users.

Usage

```r
tk_ts_.data.frame(data, select, start, end, frequency, deltat, ts.eps, silent)
tk_ts_.default(data, select, start, end, frequency, deltat, ts.eps, silent)
```

```r
tk_zooreg_.data.frame(
    data,
    select,
    date_var,
    start,
    end,
    frequency,
    deltat,
    ts.eps,
    order.by,
    silent
)
```

```r
tk_zooreg_.default(
    data,
    select,
    date_var,
    start,
    end,
    frequency,
    deltat,
    ts.eps,
    order.by,
    silent
)
```

Arguments

- `data` A time-based tibble or time-series object.
- `select` Applicable to tibbles and data frames only. The column or set of columns to be coerced to zooreg class.
- `start` the time of the first observation. Either a single number or a vector of two integers, which specify a natural time unit and a (1-based) number of samples into the time unit.
end the time of the last observation, specified in the same way as start.
frequency the number of observations per unit of time.
deltat the fraction of the sampling period between successive observations; e.g., 1/12 for monthly data. Only one of frequency or deltat should be provided.
ts.eps time series comparison tolerance. Frequencies are considered equal if their absolute difference is less than ts.eps.
silent Used to toggle printing of messages and warnings.
date.var Applicable to tibbles and data frames only. Column name to be used to order.by. NULL by default. If NULL, function will find the date or date-time column.
order.by a vector by which the observations in x are ordered. If this is specified the arguments start and end are ignored and zoo(data,order.by,frequency) is called. See zoo for more information.

---

**tk_augment_timeseries**  Augment the time series signature to the data

**Description**

Augment the time series signature to the data

**Usage**

```r
tk_augment_timeseries_signature(data)
```

**Arguments**

- `data` A time-based tibble or time-series object.

**Details**

tk_augment_timeseries_signature adds the time series signature features including numeric value, differences, year, month, day, day of week, day of month, day of year, hour, minute, second to the input data.

**Value**

Returns a tibble object describing the timeseries.

**See Also**

- `tk_get_timeseries_signature()`, `tk_get_timeseries_summary()`
Examples

```r
library(dplyr)
library(tidyquant)
library(timetk)

FANG %>%
  filter(symbol == "FB") %>%
  tk_augment_timeseries_signature()
```

---

**tk_get_timeseries**  
Get summary attributes from a time-series index

**Description**

Get summary attributes from a time-series index

**Usage**

```r
tk_get_timeseries_signature(idx)
tk_get_timeseries_summary(idx)
```

**Arguments**

*idx*  
A time-series index that is a vector of dates or datetimes.

**Details**

`tk_get_timeseries_signature` decomposes the timeseries into commonly needed features such as numeric value, differences, year, month, day, day of week, day of month, day of year, hour, minute, second.

`tk_get_timeseries_summary` returns the summary returns the start, end, units, scale, and a “summary” of the timeseries differences in seconds including the minimum, 1st quartile, median, mean, 3rd quartile, and maximum frequency. The timeseries differences give the user a better picture of the index frequency so the user can understand the level of regularity or irregularity. A perfectly regular time series will have equal values in seconds for each metric. However, this is not often the case.

**Important Note**: These functions only work with time-based indexes in datetime, date, yearmon, and yearqtr values. Regularized dates cannot be decomposed.

**Value**

Returns a tibble object describing the timeseries.

**See Also**

`tk_index()`, `tk_augment_timeseries_signature()`, `tk_make_future_timeseries()`
Examples

library(dplyr)
library(tidyquant)
library(timetk)

# Works with time-based tibbles
FB_tbl <- FANG %>% filter(symbol == "FB")
FB_idx <- tk_index(FB_tbl)

tk_get_timeseries_signature(FB_idx)
tk_get_timeseries_summary(FB_idx)

# Works with dates in any periodicity
idx_weekly <- seq.Date(from = ymd("2016-01-01"), by = 'week', length.out = 6)

tk_get_timeseries_signature(idx_weekly)
tk_get_timeseries_summary(idx_weekly)

# Works with zoo yearmon and yearqtr classes
idx_yearmon <- seq.Date(from = ymd("2016-01-01"),
                        by = "month",
                        length.out = 12) %>%
                        as.yearmon()

tk_get_timeseries_signature(idx_yearmon)
tk_get_timeseries_summary(idx_yearmon)

---

**tk_get_timeseries_unit_frequency**  
*Get the timeseries unit frequency for the primary time scales*

Description

Get the timeseries unit frequency for the primary time scales

Usage

`tk_get_timeseries_unit_frequency()`

Value

`tk_get_timeseries_unit_frequency` returns a tibble containing the timeseries frequencies in seconds for the primary time scales including "sec", "min", "hour", "day", "week", "month", "quarter", and "year".
Examples

    tk_get_timeseries_unit_frequency()

tk_get_timeseries_variables

Get date or datetime variables (column names)

Description

Get date or datetime variables (column names)

Usage

    tk_get_timeseries_variables(data)

Arguments

    data An object of class data.frame

Details

    tk_get_timeseries_variables returns the column names of date or datetime variables in a data frame. Classes that meet criteria for return include those that inherit POSIXt, Date, zoo::yearmon, zoo::yearqtr. Function was adapted from padr:::get_date_variables(). See padr helpers.R

Value

    tk_get_timeseries_variables returns a vector containing column names of date-like classes.

Examples

    library(tidyquant)
    library(timetk)

    FANG %>%
    tk_get_timeseries_variables()
tk_index

Extract an index of date or datetime from time series objects, models, forecasts

Description

Extract an index of date or datetime from time series objects, models, forecasts

Usage

```r
tk_index(data, timetk_idx = FALSE, silent = FALSE)

has_timetk_idx(data)
```

Arguments

- `data`: A time-based tibble, time-series object, time-series model, or forecast object.
- `timetk_idx`: If `timetk_idx` is `TRUE` a timetk time-based index attribute is attempted to be returned. If `FALSE` the default index is returned. See discussion below for further details.
- `silent`: Used to toggle printing of messages and warnings.

Details

`tk_index()` is used to extract the date or datetime index from various time series objects, models and forecasts. The method can be used on `tbl`, `xts`, `zoo`, `zoo_reg`, and `ts` objects. The method can additionally be used on forecast objects and a number of objects generated by modeling functions such as `Arima`, `ets`, and `HoltWinters` classes to get the index of the underlying data.

The boolean `timetk_idx` argument is applicable to regularized time series objects such as `ts` and `zoo_reg` classes that have both a regularized index and potentially a "timetk index" (a time-based attribute). When set to `FALSE` the regularized index is returned. When set to `TRUE` the time-based timetk index is returned if present.

`has_timetk_idx()` is used to determine if the object has a "timetk index" attribute and can thus benefit from the `tk_index(timetk_idx = TRUE)`. `TRUE` indicates the "timetk index" attribute is present. `FALSE` indicates the "timetk index" attribute is not present. If `FALSE`, the `tk_index()` function will return the default index for the data type.

**Important Note:** To gain the benefit of `timetk_idx` the time series must have a timetk index. Use `has_timetk_idx` to determine if the object has a timetk index. This is particularly important for `ts` objects, which by default do not contain a time-based index and therefore must be coerced from time-based objects such as `tbl`, `xts`, or `zoo` using the `tk_ts()` function in order to get the "timetk index" attribute. Refer to `tk_ts()` for creating persistent date / datetime index during coercion to `ts`.

Value

Returns a vector of date or date times
tk_make_future_timeseries

Make a future time series from an existing time series

Description

Make a future time series from an existing time series

Usage

```r
tk_make_future_timeseries(
  idx,
  n_future,
  inspect_weekdays = FALSE,
  inspect_months = FALSE,
  skip_values = NULL,
  insert_values = NULL
)
```
**tk_make_future_timeseries**

**Arguments**

- **idx**: A vector of dates
- **n_future**: Number of future observations
- **inspect_weekdays**: Uses a logistic regression algorithm to inspect whether certain weekdays (e.g. weekends) should be excluded from the future dates. Default is FALSE.
- **inspect_months**: Uses a logistic regression algorithm to inspect whether certain days of months (e.g. last two weeks of year or seasonal days) should be excluded from the future dates. Default is FALSE.
- **skip_values**: A vector of same class as idx of timeseries values to skip.
- **insert_values**: A vector of same class as idx of timeseries values to insert.

**Details**

**tk_make_future_timeseries** returns a time series based on the input index frequency and attributes.

The argument n_future determines how many future index observations to compute.

The inspect_weekdays and inspect_months arguments apply to "daily" (scale = "day") data (refer to tk_get_timeseries_summary() to get the index scale). The inspect_weekdays argument is useful in determining missing days of the week that occur on a weekly frequency such as every week, every other week, and so on. It’s recommended to have at least 60 days to use this option.

The inspect_months argument is useful in determining missing days of the month, quarter or year; however, the algorithm can inadvertently select incorrect dates if the pattern is erratic. For example, some holidays do not occur on the same day of each month, and as a result the incorrect day may be selected in certain years. It’s recommended to always review the date results to ensure the future days match the user’s expectations. It’s recommended to have at least two years of days to use this option.

The skip_values and insert_values arguments can be used to remove and add values into the series of future times. The values must be the same format as the idx class. The skip_values argument useful for passing holidays or special index values that should be excluded from the future time series. The insert_values argument is useful for adding values back that the algorithm may have excluded.

**Value**

A vector containing future dates

**See Also**

- **tk_index()**, **tk_get_timeseries_summary()**, **tk_get_timeseries_signature()**

**Examples**

```r
library(dplyr)
library(tidyquant)
library(timetk)
```
# Basic example
idx <- c("2016-01-01 00:00:00", "2016-01-01 00:00:03", "2016-01-01 00:00:06") %>% ymd_hms()

# Make next three dates in series
idx %>%
  tk_make_future_timeseries(n_future = 3)

# Create index of days that FB stock will be traded in 2017 based on 2016 + holidays
FB_tbl <- FANG %>% filter(symbol == "FB")

# Remove holidays with skip_values, and remove weekends with inspect_weekdays = TRUE
FB_tbl %>%
  tk_index() %>%
  tk_make_future_timeseries(n_future = 366, inspect_weekdays = TRUE, skip_values = holidays)

# Works with regularized indexes as well
c(2016.00, 2016.25, 2016.50, 2016.75) %>%
  tk_make_future_timeseries(n_future = 4)

# Works with zoo yearmon and yearqtr too
c("2016 Q1", "2016 Q2", "2016 Q3", "2016 Q4") %>%
  as.yearqtr() %>%
  tk_make_future_timeseries(n_future = 4)

---

**tk_tbl**

Coerce time-series objects to tibble.

**Description**

Coerce time-series objects to tibble.

**Usage**

```
tk_tbl(
  data,
  preserve_index = TRUE,
  rename_index = "index",
  timetk_idx = FALSE,
  silent = FALSE,
  ...
)
```
Arguments

- **data**: A time-series object.
- **preserve_index**: Attempts to preserve a time series index. Default is `TRUE`.
- **rename_index**: Enables the index column to be renamed.
- **timetk_idx**: Used to return a date / datetime index for regularized objects that contain a timetk "index" attribute. Refer to `tk_index()` for more information on returning index information from regularized timeseries objects (i.e. ts).
- **silent**: Used to toggle printing of messages and warnings.
- ... Additional parameters passed to the `tibble::as_tibble()` function.

Details

`tk_tbl` is designed to coerce time series objects (e.g. xts, zoo, ts, timeSeries, etc) to tibble objects. The main advantage is that the function keeps the date / date-time information from the underlying time-series object.

When `preserve_index = TRUE` is specified, a new column, `index`, is created during object coercion, and the function attempts to preserve the date or date-time information. The date / date-time column name can be changed using the `rename_index` argument.

The `timetk_idx` argument is applicable when coercing ts objects that were created using `tk_ts()` from an object that had a time base (e.g. tbl, xts, zoo). Setting `timetk_idx = TRUE` enables returning the timetk "index" attribute if present, which is the original (non-regularized) time-based index.

Value

Returns a tibble object.

See Also

- `tk_xts()`, `tk_zoo()`, `tk_zooreg()`, `tk_ts()`

Examples

```r
library(tidyverse)
library(timetk)

data_tbl <- tibble(
  date = seq.Date(from = as.Date("2010-01-01"), by = 1, length.out = 5),
  x    = seq(100, 120, by = 5)
)

### ts to tibble: Comparison between as.data.frame() and tk_tbl()
data_ts <- tk_ts(data_tbl, start = c(2010,1), freq = 365)

# No index
as.data.frame(data_ts)
```
tk_ts

Coerce time series objects and tibbles with date/date-time columns to ts.

Description

Coerce time series objects and tibbles with date/date-time columns to ts.
tk_ts

Usage

```r
tk_ts(
  data,
  select = NULL,
  start = 1,
  end = numeric(),
  frequency = 1,
  deltat = 1,
  ts.eps = getOption("ts.eps"),
  silent = FALSE
)
```

Arguments

- `data`: A time-based tibble or time-series object.
- `select`: Applicable to tibbles and data frames only. The column or set of columns to be coerced to `ts` class.
- `start`: the time of the first observation. Either a single number or a vector of two integers, which specify a natural time unit and a (1-based) number of samples into the time unit. See the examples for the use of the second form.
- `end`: the time of the last observation, specified in the same way as `start`.
- `frequency`: the number of observations per unit of time.
- `deltat`: the fraction of the sampling period between successive observations; e.g., 1/12 for monthly data. Only one of `frequency` or `deltat` should be provided.
- `ts.eps`: time series comparison tolerance. Frequencies are considered equal if their absolute difference is less than `ts.eps`.
- `silent`: Used to toggle printing of messages and warnings.

Details

tk_ts() is a wrapper for stats::ts() that is designed to coerce tibble objects that have a "time-base" (meaning the values vary with time) to ts class objects. There are two main advantages:

1. Non-numeric columns get removed instead of being populated by NA's.
2. The returned ts object retains a "timetk index" (and various other attributes) if detected. The "timetk index" can be used to coerce between tbl, xts, zoo, and ts data types.
The `select` argument is used to select subsets of columns from the incoming data.frame. Only columns containing numeric data are coerced. At a minimum, a frequency and a start should be specified.

For non-data.frame object classes (e.g., `xts`, `zoo`, `timeSeries`, etc) the objects are coerced using `stats::ts()`.

`tk_ts_` is a nonstandard evaluation method.

**Value**

Returns a `ts` object.

**See Also**

`tk_index()`, `tk_tbl()`, `tk_xts()`, `tk_zoo()`, `tk_zooreg()`

**Examples**

```r
library(tidyverse)
library(timetk)

### tibble to ts: Comparison between `tk_ts()` and `stats::ts()`
data_tbl <- tibble::tibble(
  date = seq.Date(as.Date("2016-01-01"), by = 1, length.out = 5),
  x   = rep("chr values", 5),
  y   = cumsum(1:5),
  z   = cumsum(11:15) * rnorm(1))

# as.ts: Character columns introduce NA's; Result does not retain index
stats::ts(data_tbl[-1], start = 2016)

# tk_ts: Only numeric columns get coerced; Result retains index in numeric format
data_ts <- tk_ts(data_tbl, start = 2016)
data_ts

# timetk index
tk_index(data_ts, timetk_idx = FALSE)  # Regularized index returned
tk_index(data_ts, timetk_idx = TRUE)   # Original date index returned

# Coerce back to tibble
data_ts %>% tk_tbl(timetk_idx = TRUE)

### Using select

```

```
tk_xts

Coerce time series objects and tibbles with date/date-time columns to xts.

Description

Coerce time series objects and tibbles with date/date-time columns to xts.

Usage

tk_xts(data, select = NULL, date_var = NULL, silent = FALSE, ...)
tk_xts_(data, select = NULL, date_var = NULL, silent = FALSE, ...)  

Arguments

data

A time-based tibble or time-series object.

select

Applicable to tibbles and data frames only. The column or set of columns to be coerced to ts class.

date_var

Applicable to tibbles and data frames only. Column name to be used to order.by. NULL by default. If NULL, function will find the date or date-time column.

silent

Used to toggle printing of messages and warnings.

... Additional parameters to be passed to xts::xts(). Refer to xts::xts().

Details

tk_xts is a wrapper for xts::xts() that is designed to coerce tibble objects that have a "time-base" (meaning the values vary with time) to xts class objects. There are three main advantages:

1. Non-numeric columns that are not removed via select are dropped and the user is warned. This prevents an error or coercion issue from occurring.

2. The date column is auto-detected if not specified by date_var. This takes the effort off the user to assign a date vector during coercion.

3. ts objects are automatically coerced if a "timetk index" is present. Refer to tk_ts().

The select argument can be used to select subsets of columns from the incoming data.frame. Only columns containing numeric data are coerced. The date_var can be used to specify the column with the date index. If date_var = NULL, the date / date-time column is interpreted. Optionally, the order.by argument from the underlying xts::xts() function can be used. The user must pass a vector of dates or date-times if order.by is used.

For non-data.frame object classes (e.g. xts, zoo, timeSeries, etc) the objects are coerced using xts::xts().

tk_xts_ is a nonstandard evaluation method.
tk_zoo

**Value**

Returns a xts object.

**See Also**

`tk_tbl()`, `tk_zoo()`, `tk_zooreg()`, `tk_ts()`

**Examples**

```r
library(tidyverse)
library(timetk)

### tibble to xts: Comparison between tk_xts() and xts::xts()
data_tbl <- tibble::tibble(
  date = seq.Date(as.Date("2016-01-01"), by = 1, length.out = 5),
  x   = rep("chr values", 5),
  y   = cumsum(1:5),
  z   = cumsum(11:15) * rnorm(1))
# xts: Character columns cause coercion issues; order.by must be passed a vector of dates
xts::xts(data_tbl[, -1], order.by = data_tbl$date)
# tk_xts: Non-numeric columns automatically dropped; No need to specify date column
tk_xts(data_tbl)

# ts can be coerced back to xts
data_tbl %>%
  tk_ts(start = 2016, freq = 365) %>%
  tk_xts()

### Using select and date_var
tk_xts(data_tbl, select = y, date_var = date)

### NSE: Enables programming
date_var <- "date"
select <- "y"
tk_xts_(data_tbl, select = select, date_var = date_var)
```

---

**tk_zoo**

**Coerce time series objects and tibbles with date/date-time columns to xts.**

**Description**

Coerce time series objects and tibbles with date/date-time columns to xts.
tk_zoo

Usage

\[
\text{tk_zoo}(\text{data, select = NULL, date_var = NULL, silent = FALSE, } \ldots )
\]

\[
\text{tk_zoo_}(\text{data, select = NULL, date_var = NULL, silent = FALSE, } \ldots )
\]

Arguments

- **data**: A time-based tibble or time-series object.
- **select**: Applicable to tibbles and data frames only. The column or set of columns to be coerced to \texttt{ts} class.
- **date_var**: Applicable to tibbles and data frames only. Column name to be used to order.by. NULL by default. If NULL, the function will find the date or date-time column.
- **silent**: Used to toggle printing of messages and warnings.
- **...**: Additional parameters to be passed to \texttt{xts::xts()}. Refer to \texttt{xts::xts()}.

Details

tk_zoo is a wrapper for \texttt{zoo::zoo()} that is designed to coerce tibble objects that have a "time-base" (meaning the values vary with time) to zoo class objects. There are three main advantages:

1. Non-numeric columns that are not removed via \texttt{select} are dropped and the user is warned. This prevents an error or coercion issue from occurring.
2. The date column is auto-detected if not specified by \texttt{date_var}. This takes the effort off the user to assign a date vector during coercion.
3. \texttt{ts} objects are automatically coerced if a "timetk index" is present. Refer to \texttt{tk_ts()}.

The \texttt{select} argument can be used to select subsets of columns from the incoming data.frame. Only columns containing numeric data are coerced. The \texttt{date_var} can be used to specify the column with the date index. If \texttt{date_var = NULL}, the date / date-time column is interpreted. Optionally, the \texttt{order.by} argument from the underlying \texttt{zoo::zoo()} function can be used. The user must pass a vector of dates or date-times if \texttt{order.by} is used. \textit{Important Note: The \ldots arguments are passed to \texttt{xts::xts()}, which enables additional information (e.g. time zone) to be an attribute of the zoo object.} For non-data.frame object classes (e.g. \texttt{xts}, \texttt{zoo}, \texttt{timeSeries}, etc) the objects are coerced using \texttt{zoo::zoo()}.

\texttt{tk_zoo_} is a nonstandard evaluation method.

Value

Returns a \texttt{zoo} object.

See Also

\texttt{tk_tbl()}, \texttt{tk_xts()}, \texttt{tk_zooreg()}, \texttt{tk_ts()}
### Examples

```r
library(tidyverse)
library(timetk)

### tibble to zoo: Comparison between tk_zoo() and zoo::zoo()

data_tbl <- tibble::tibble(
  date = seq.Date(as.Date("2016-01-01"), by = 1, length.out = 5),
  x = rep("chr values", 5),
  y = cumsum(1:5),
  z = cumsum(11:15) * rnorm(1))

# zoo: Characters will cause error; order.by must be passed a vector of dates
zoo::zoo(data_tbl[-c(1,2)], order.by = data_tbl$date)

# tk_zoo: Character columns dropped with a warning; No need to specify dates (auto detected)
tk_zoo(data_tbl)

# ts can be coerced back to zoo
data_tbl %>%
  tk_ts(start = 2016, freq = 365) %>%
  tk_zoo()

### Using select and date_var

tk_zoo(data_tbl, select = y, date_var = date)

### NSE: Enables programming

date_var <- "date"
select <- "y"
tk_zoo_(data_tbl, select = select, date_var = date_var)
```

---

**tk_zooreg**

Coerce time series objects and tibbles with date/date-time columns to **ts**.

---

**Description**

Coerce time series objects and tibbles with date/date-time columns to ts.

**Usage**

```r
tk_zooreg(  
data,  
  select = NULL,  
  date_var = NULL,  
  start = 1,  
  end = numeric(),
)```
tk_zooreg

frequency = 1,
deltat = 1,
ts.eps = getOption("ts.eps"),
order.by = NULL,
silent = FALSE
)

tk_zooreg_(
  data,
  select = NULL,
  date_var = NULL,
  start = 1,
  end = numeric(),
  frequency = 1,
  deltat = 1,
  ts.eps = getOption("ts.eps"),
  order.by = NULL,
  silent = FALSE
)

Arguments

data A time-based tibble or time-series object.
select Applicable to tibbles and data frames only. The column or set of columns to be coerced to zooreg class.
date_var Applicable to tibbles and data frames only. Column name to be used to order.by. NULL by default. If NULL, function will find the date or date-time column.
start the time of the first observation. Either a single number or a vector of two integers, which specify a natural time unit and a (1-based) number of samples into the time unit.
end the time of the last observation, specified in the same way as start.
frequency the number of observations per unit of time.
deltat the fraction of the sampling period between successive observations; e.g., 1/12 for monthly data. Only one of frequency or deltat should be provided.
ts.eps time series comparison tolerance. Frequencies are considered equal if their absolute difference is less than ts.eps.
order.by a vector by which the observations in x are ordered. If this is specified the arguments start and end are ignored and zoo(data,order.by,frequency) is called. See zoo for more information.
silent Used to toggle printing of messages and warnings.

Details

tk_zooreg() is a wrapper for zoo::zooreg() that is designed to coerce tibble objects that have a “time-base” (meaning the values vary with time) to zooreg class objects. There are two main advantages:
1. Non-numeric columns get removed instead causing coercion issues.

2. If an index is present, the returned zooreg object retains an index retrievable using `tk_index()`.

The `select` argument is used to select subsets of columns from the incoming data.frame. The `date_var` can be used to specify the column with the date index. If `date_var = NULL`, the date / date-time column is interpreted. Optionally, the `order.by` argument from the underlying `xts::xts()` function can be used. The user must pass a vector of dates or date-times if `order.by` is used. Only columns containing numeric data are coerced. *At a minimum, a frequency and a start should be specified.*

For non-data.frame object classes (e.g. `xts`, `zoo`, `timeSeries`, etc) the objects are coerced using `zoo::zooreg()`.

`tk_zooreg_` is a nonstandard evaluation method.

**Value**

Returns a zooreg object.

**See Also**

`tk_tbl()`, `tk_xts()`, `tk_zoo()`, `tk_ts()`

**Examples**

```r
### tibble to zooreg: Comparison between tk_zooreg() and zoo::zooreg()
data_tbl <- tibble::tibble(
  date = seq.Date(as.Date("2016-01-01"), by = 1, length.out = 5),
  x = rep("chr values", 5),
  y = cumsum(1:5),
  z = cumsum(11:15) * rnorm(1))

# zoo::zooreg: Values coerced to character; Result does not retain index
data_zooreg <- zoo::zooreg(data_tbl[-1], start = 2016, freq = 365)
rownames(data_zooreg) # NULL, no dates retained

data_tk_zooreg <- tk_zooreg(data_tbl, start = 2016, freq = 365)
# No inadvertent coercion to character class

# timetk index
tk_index(data_tk_zooreg, timetk_idx = TRUE) # Original date index returned

### Using select and date_var
tk_zooreg(data_tbl, select = y, date_var = date, start = 2016, freq = 365)

### NSE: Enables programming
select <- "y"
date_var <- "date"
tk_zooreg_(data_tbl, select = select, date_var = date_var, start = 2016, freq = 365)
```
Index

*Topic datagen

  step_timeseries_signature, 2

has_timetk_idx (tk_index), 10

recipes::bake.recipe(), 3
recipes::prep.recipe(), 3
recipes::recipe(), 3
recipes::selections(), 3
recipes::step_date(), 3
recipes::step_holiday(), 3
recipes::step_normalize(), 3
recipes::step_rm(), 3

step_timeseries_signature, 2

	ibble::as_tibble(), 14
tidy.step_timeseries_signature

  (step_timeseries_signature), 2

timetk, 4
timetk_internal, 5
tk_augment_timeseries, 6
tk_augment_timeseries_signature

  (tk_augment_timeseries), 6
tk_augment_timeseries_signature(), 7
tk_get_timeseries, 7

	tk_get_timeseries_signature

  (tk_get_timeseries), 7
tk_get_timeseries_signature(), 6, 12

tk_get_timeseries_summary

  (tk_get_timeseries), 7
tk_get_timeseries_summary(), 6, 12
tk_get_timeseries_unit_frequency, 8
tk_get_timeseries_variables, 9
tk_index, 10


tk_index(), 7, 12, 14, 17, 23
tk_make_future_timeseries, 11
tk_make_future_timeseries(), 7
tk_tbl, 13
tk_tbl(), 11, 17, 19, 20, 23

  tk_ts, 15
tk_ts(), 10, 11, 14, 18–20, 23
tk_ts_(tk_ts), 15
tk_ts_.data.frame (timetk_internal), 5
tk_ts_.default (timetk_internal), 5
tk_xts, 18
tk_xts(), 11, 14, 17, 20, 23
tk_xts_(tk_xts), 18
tk_zoo, 19
tk_zoo(), 11, 14, 17, 19, 23
tk_zoo_(tk_zoo), 19
tk_zooreg, 21
tk_zooreg(), 11, 14, 17, 19, 20
tk_zooreg_(tk_zooreg), 21
tk_zooreg_.data.frame

  (timetk_internal), 5
tk_zooreg_.default (timetk_internal), 5

zoo, 6, 22