Package ‘csalert’
June 24, 2024

Title  Alerts from Public Health Surveillance Data
Version  2024.6.24
Description  Helps create alerts and determine trends by using various methods to analyze public health surveillance data. The primary analysis method is based upon a published analytics strategy by Benedetti (2019) <doi:10.5588/pha.19.0002>.
Depends  R (>= 3.3.0)
License  MIT + file LICENSE
BugReports  https://github.com/csids/csalert/issues
Encoding  UTF-8
Imports  data.table, magrittr, ggplot2, glm2, cstidy, cstime, lubridate, stringr, surveillance
Suggests  testthat, knitr, rmarkdown, rstudioapi, glue, covidnor, cdata, csmaps, ggrepel, plnr
RoxygenNote  7.2.3
VignetteBuilder  knitr
NeedsCompilation  no
Author  Beatriz Valcarcel Salamanca [aut],
        Chi Zhang [aut] (<https://orcid.org/0000-0003-0501-5909>),
        Richard Aubrey White [aut, cre]
        (<https://orcid.org/0000-0002-6747-1726>),
        CSIDS [cph]
Maintainer  Richard Aubrey White <hello@rwhite.no>
Repository  CRAN
Date/Publication  2024-06-24 12:20:12 UTC

Contents

add_holiday_effect .................................................. 2
prediction_interval .................................................. 2
add_holiday_effect  

**Holiday effect** ——

**Description**

The effect of public holiday on a time series of daily counts

**Usage**

```r
add_holiday_effect(data, holiday_data, holiday_effect = 2)
```

**Arguments**

- `data`: A `csfmt_rds` data object
- `holiday_data`: dates
- `holiday_effect`: Ending date of the simulation period.

**Value**

A `csfmt_rts_data_v1`, `data.table` containing

---

prediction_interval  

**Prediction thresholds**

**Description**

Prediction thresholds

**Usage**

```r
prediction_interval(object, newdata, alpha = 0.05, z = NULL, ...)
```
Arguments

- **object**: Object
- **newdata**: New data
- **alpha**: Two-sided alpha (e.g. 0.05)
- **z**: Similar to alpha (e.g. z=1.96 is the same as alpha=0.05)
- **...**: dots

---

Prediction thresholds

Description

Prediction thresholds

Usage

```r
## S3 method for class 'glm'
prediction_interval(
  object,  # Object
  newdata,  # New data
  alpha = 0.05,  # Two-sided alpha (e.g 0.05)
  z = NULL,  # Similar to alpha (e.g. z=1.96 is the same as alpha=0.05)
  skewness_transform = "none",  # "none", "1/2", "2/3"
  ...)  # dots
```

Arguments

- **object**: Object
- **newdata**: New data
- **alpha**: Two-sided alpha (e.g 0.05)
- **z**: Similar to alpha (e.g. z=1.96 is the same as alpha=0.05)
- **skewness_transform**: "none", "1/2", "2/3"
- **...**: dots
**short_term_trend**

*Determine the short term trend of a timeseries*

**Description**


**Usage**

```
short_term_trend(x, ...)  
## S3 method for class 'csfmt_rts_data_v1'
short_term_trend(
  x,  
  numerator,  
  denominator = NULL,  
  prX = 100,  
  trend_isoyearweeks = 6,  
  remove_last_isoyearweeks = 0,  
  forecast_isoyearweeks = trend_isoyearweeks,  
  numerator_naming_prefix = "from_numerator",  
  denominator_naming_prefix = "from_denominator",  
  statistics_naming_prefix = "universal",  
  remove_training_data = FALSE,  
  include_decreasing = FALSE,  
  alpha = 0.05,  
  ...  
)
```

**Arguments**

- `x`: Data object
- `...`: Not in use.
- `numerator`: Character of name of numerator
- `denominator`: Character of name of denominator (optional)
- `prX`: If using denominator, what scaling factor should be used for numerator/denominator?
- `trend_isoyearweeks`: Same as trend_dates, but used if granularity_geo=='isoyearweek'
- `remove_last_isoyearweeks`: Same as remove_last_dates, but used if granularity_geo=='isoyearweek'
- `forecast_isoyearweeks`: Same as forecast_dates, but used if granularity_geo=='isoyearweek'
- `numerator_naming_prefix`: "from_numerator", "generic", or a custom prefix
**short_term_trend_sts_v1**

- **denominator_naming_prefix**
  - "from_denominator", "generic", or a custom prefix

- **statistics_naming_prefix**
  - "universal" (one variable for trend status, one variable for doubling dates), "from_numerator_and_prX" (If denominator is NULL, then one variable corresponding to numerator. If denominator exists, then one variable for each of the prXs)

- **remove_training_data**
  - Boolean. If TRUE, removes the training data (i.e. 1:(trend_dates-1) or 1:(trend_isoyearweeks-1)) from the returned dataset.

- **include_decreasing**
  - If true, then *_trend*_status contains the levels c("training", "forecast", "decreasing", "null", "increasing"), otherwise the levels c("training", "forecast", "notincreasing", "increasing").

- **alpha**
  - Significance level for change in trend.

**Value**

The original csfmt_rts_data_v1 dataset with extra columns. *trend*_status contains a factor with levels c("training", "forecast", "decreasing", "null", "increasing"), while *doublingdays* contains the expected number of days before the numerator doubles.

**Examples**

```r
# Load the datasets
library(cstidy)
library(csalert)

d <- cstidy::nor_covid19_icu_and_hospitalization_csfmt_rts_v1
d <- d[granularity_time=="isoyearweek"]
res <- csalert::short_term_trend(
  d,
  numerator = "hospitalization_with_covid19_as_primary_cause_n",
  trend_isoyearweeks = 6
)
print(res[, .(isoyearweek,
          hospitalization_with_covid19_as_primary_cause_n,
          hospitalization_with_covid19_as_primary_cause_trend0_41_status)])
```

**Description**

The method is based upon a published analytics strategy by Benedetti (2019) <doi:10.5588/pha.19.0002>. This function has been frozen on 2024-06-24. It is designed to use sts

**Usage**

```r
short_term_trend_sts_v1(sts, control = list(w = 5, alpha = 0.05))
```
Arguments

- `sts`: Data object of type sts.
- `control`: Control object, a named list with several elements.
  - `w`: Length of the window that is being analyzed.
  - `alpha`: Significance level for change in trend.

Value

sts object with the alarms slot set to 0/1 if not-increasing/increasing.

Examples

```r
d <- cstidy::nor_covid19_icu_and_hospitalization_csfmt_rts_v1
d <- d[, granularity_time="isoyearweek"]
sts <- surveillance::sts(
  observed = d$hospitalization_with_covid19_as_primary_cause_n, # weekly number of cases
  start = c(d$isoyear[1], d$isoweek[1]), # first week of the time series
  frequency = 52
)
x <- csalert::short_term_trend_sts_v1(
  sts,
  control = list(
    w = 5,
    alpha = 0.05
  )
)
plot(x)
```

---

**signal_detection_hlm**  
*Determine the short term trend of a timeseries*

Description


Usage

```r
signal_detection_hlm(x, ...)
```

```r
## S3 method for class 'csfmt_rts_data_v1'
signal_detection_hlm(
  x,
  value,
  baseline_isoyears = 5,
  remove_last_isoyearweeks = 0,
  forecast_isoyearweeks = 2,
  value_naming_prefix = "from_numerator",
)```
Arguments

- **x**
  Data object

- **value**
  Character of name of value

- **baseline_isoyears**
  Number of years in the past you want to include as baseline

- **remove_last_isoyearweeks**
  Number of isoyearweeks you want to remove at the end (due to unreliable data)

- **forecast_isoyearweeks**
  Number of isoyearweeks you want to forecast into the future

- **value_naming_prefix**
  "from_numerator", "generic", or a custom prefix

- **remove_training_data**
  Boolean. If TRUE, removes the training data (i.e. 1:(trend_isoyearweeks-1)) from the returned dataset.

Value

The original csfmt_rts_data_v1 dataset with extra columns. *trend* status contains a factor with levels c("training", "forecast", "decreasing", "null", "increasing"), while *doublingdays* contains the expected number of days before the numerator doubles.

Examples

```r
d <- cstidy::nor_covid19_icu_and_hospitalization_csfmt_rts_v1
d <- d[granularity_time=="isoyearweek"]
res <- csalert::signal_detection_hlm(d,
  value = "hospitalization_with_covid19_as_primary_cause_n",
  baseline_isoyears = 1)
print(res[, .(isoyearweek,
  hospitalization_with_covid19_as_primary_cause_n,
  hospitalization_with_covid19_as_primary_cause_forecasted_n,
  hospitalization_with_covid19_as_primary_cause_forecasted_n_forecast,
  hospitalization_with_covid19_as_primary_cause_baseline_predinterval_q50x0_n,
  hospitalization_with_covid19_as_primary_cause_baseline_predinterval_q99x5_n,
  hospitalization_with_covid19_as_primary_cause_n_status)])
```
simulate_baseline_data

Simulate baseline data — Simulation of baseline data.

Description

This function simulates a time series of daily counts in the absence of outbreaks. Data is simulated using a poisson/negative binomial model as described in Noufaily et al. (2019). Properties of time series such as frequency of baseline observations, trend, seasonal and weekly pattern can be specified in the simulation.

Usage

```
simulate_baseline_data(
    start_date,
    end_date,
    seasonal_pattern_n,  
    weekly_pattern_n, 
    alpha, 
    beta,  
    gamma_1,  
    gamma_2, 
    gamma_3, 
    gamma_4,  
    phi, 
    shift_1 
)
```

Arguments

- `start_date`: Starting date of the simulation period. Date is in the format of 'yyyy-mm-dd'.
- `end_date`: Ending date of the simulation period. Date is in the format of 'yyyy-mm-dd'.
- `seasonal_pattern_n`: Number of seasonal patterns. For no seasonal pattern `seasonal_pattern_n = 0`. `Seasonal_pattern_n = 1` represents annual pattern. `Seasonal_pattern_n = 2` indicates biannual pattern.
- `weekly_pattern_n`: Number of weekly patterns. For no specific weekly pattern, `weekly_pattern_n = 0`. `Weekly_pattern_n = 1` represents one weekly peak.
- `alpha`: The parameter is used to specify the baseline frequencies of reports.
- `beta`: The parameter is used to specify linear trend.
- `gamma_1`: The parameter is used to specify the seasonal pattern.
- `gamma_2`: The parameter is used to specify the seasonal pattern.
- `gamma_3`: The parameter is used to specify day-of-the week pattern.
gamma_4  The parameter is used to specify day-of-the week pattern  
phi  Dispersion parameter. If phi =0, a Poisson model is used to simulate baseline data.  
shift_1  Horizontal shift parameter to help control over week/month peaks.

Value

A csfmt_rts_data_v1, data.table containing a time series of counts

Examples

```
baseline <- simulate_baseline_data(
  start_date = as.Date("2012-01-01"),
  end_date = as.Date("2019-12-31"),
  seasonal_pattern_n = 1,
  weekly_pattern_n = 1,
  alpha = 3,
  beta = 0,
  gamma_1 = 0.8,
  gamma_2 = 0.6,
  gamma_3 = 0.8,
  gamma_4 = 0.4,
  phi = 4,
  shift_1 = 29 )
```

Description

Simulation of seasonal outbreaks for syndromes/diseases that follows seasonal trends. Seasonal outbreaks are more variable both in size and timing than seasonal patterns. The number of seasonal outbreaks occur in a year are defined by n_season_outbreak. The parameters week_season_start and week_season_end define the season window. The start of the seasonal outbreak is drawn from the season window weeks, with higher probability of outbreak occurs around the peak of the season (week_season_peak). The seasonal outbreak size (excess number of cases that occurs during the outbreak) is simulated using a poisson distribution as described in Noufaily et al. (2019).

Usage

```
simulate_seasonal_outbreak_data(data,
  week_season_start = 40,
  week_season_peak = 4,
```
simulate_spike_outbreak_data

```r
week_season_end = 20,
n_season_outbreak = 1,
m = 50
```

**Arguments**

- **data**: A csfmt_rds data object
- **week_season_start**: Starting season week number
- **week_season_peak**: Peak of the season week number
- **week_season_end**: Ending season week number
- **n_season_outbreak**: Number of seasonal outbreaks to be simulated
- **m**: Parameter to determine the size of the outbreak (m times the standard deviation of the baseline count at the starting day of the seasonal outbreak)

**Value**

A csfmt_rts_data_v1, data.table

---

**simulate_spike_outbreak_data**

*Simulate spiked outbreaks* ---

**Description**

Simulation of spiked outbreak as described in Noufaily et al. (2019). The method for simulating spiked outbreak is similar to seasonal outbreaks simulation but they are shorter in duration and are added only the last year of data (prediction data). Spiked outbreaks can start at any week during the prediction data.

**Usage**

```r
simulate_spike_outbreak_data(data, n_sp_outbreak = 1, m)
```

**Arguments**

- **data**: A csfmt_rds data object
- **n_sp_outbreak**: Number of spiked outbreaks to be simulated
- **m**: Parameter to determine the size of the outbreak (m times the standard deviation of the baseline count at the starting day of the seasonal outbreak)

**Value**

A csfmt_rts_data_v1, data.table
Index

add_holiday_effect, 2
prediction_interval, 2
prediction_interval.glm, 3
short_term_trend, 4
short_term_trend_sts_v1, 5
signal_detection_hlm, 6
simulate_baseline_data, 8
simulate_seasonal_outbreak_data, 9
simulate_spike_outbreak_data, 10