Package ‘bbdetection’

April 1, 2021

Type Package
Title Identification of Bull and Bear States of the Market
Version 1.0
Author Valeriy Zakamulin
Maintainer Valeriy Zakamulin <valeriz@uia.no>
Description Implements two algorithms of detecting Bull and Bear markets in stock prices: the algorithm of Pagan and Sossounov (2002, <doi:10.1002/jae.664>) and the algorithm of Lunde and Timmermann (2004, <doi:10.1198/073500104000000136>). The package also contains functions for printing out the dating of the Bull and Bear states of the market, the descriptive statistics of the states, and functions for plotting the results. For the sake of convenience, the package includes the monthly and daily data on the prices (not adjusted for dividends) of the S&P 500 stock market index.
License GPL-3
Encoding UTF-8
LazyData TRUE
Depends R (>= 4.0)
Imports Rcpp (>= 0.12.5), zoo, xtable, ggplot2
LinkingTo Rcpp
RoxygenNote 7.1.1
NeedsCompilation yes
Repository CRAN
Date/Publication 2021-04-01 14:30:09 UTC

R topics documented:

  bb.dating.states ................................................... 2
  bb.plot ............................................................. 3
  bb.summary.stat .................................................. 4
  djiad ............................................................... 4
  djiam ............................................................... 5
bb.dating.states

Prints out the dating of bull-bear states

Description

This function prints out (in console window) the dating of bull-bear states. The outcome of this function is a table in LaTeX format.

Usage

bb.dating.states(price, bull, dates)

Arguments

price a numeric vector of price values
bull a logical vector that contains the states of the market. This vector is returned by function run_dating_alg or run_filtering_alg.
dates a vector of dates

Value

A data frame object that contains the dating of bull-bear states.

Examples

{
  library(zoo)
  library(xtable)
  library(ggplot2)
  sp500 <- sp500m # choose the monthly data
dates <- index(sp500) # retrieve dates
dates <- as.yearmon(dates) # convert dates to "yearmon" format if monthly data
price <- as.vector(coredata(sp500)) # retrieve prices
setpar_dating_alg(4, 6, 4, 16, 20) # parameters for monthly data
bull <- run_dating_alg(price) # detect the states
bb.dating.states(price, bull, dates)
}
bb.plot  

Plots the log of prices and highlight bear states

Description

This function plots the log of prices and highlights bear states

Usage

bb.plot(price, bull, dates, price.name=NULL, log.scale=TRUE)

Arguments

price a numeric vector of price values
bull a logical vector that contains the states of the market. This vector is returned by function run_dating_alg or run_filtering_alg.
dates a vector of dates in Date format
price.name the name of the time-series of prices that will appear on the y-axis of the plot
log.scale a logical variable that specifies whether to use log scale along the y-axis

Value

None

Examples

{
  library(zoo)
  library(xtable)
  library(ggplot2)
  price <- as.vector(coredata(sp500m)) # retrieve monthly prices
  dates <- index(sp500m) # retrieve dates from zoo-object
  setpar_dating_alg(4, 6, 5, 15, 20) # parameters for monthly data
  bull <- run_dating_alg(price) # detect bull-bear states
  bb.plot(price, bull, dates, "S&P 500") # plot the result
}
bb.summary.stat | Prints out the summary statistics of bull-bear states

Description
This function prints out (in console window) the summary statistics of bull-bear states. The outcome of this function is a table in LaTeX format.

Usage
bb.summary.stat(price, bull)

Arguments
- price: a numeric vector of price values
- bull: a logical vector that contains the states of the market. This vector is returned by function run_dating_alg or run_filtering_alg.

Value
A data frame that contains the descriptive statistics.

Examples
{
  library(zoo)
  library(xtable)
  library(ggplot2)
  sp500 <- sp500m # choose the monthly data
  price <- as.vector(coredata(sp500)) # retrieve prices
  setpar_dating_alg(4, 6, 4, 16, 20) # parameters for monthly data
  bull <- run_dating_alg(price) # detect the states
  bb.summary.stat(price, bull)
}

djiad | Daily data the Dow Jones Industrial Average index

Description
The daily closing prices of the DJIA index from January 1985

Usage
djiad
Format

An object of class "zoo" containing the daily closing prices of the DJIA index from January 1985

Source

https://finance.yahoo.com/

See Also

Other data sets: djiam, sp500d, sp500m

Description

The monthly closing prices of the DJIA index from January 1985

Usage

djiam

Format

An object of class "zoo" containing the monthly closing prices of the DJIA index from January 1985

Source

https://finance.yahoo.com/

See Also

Other data sets: djiad, sp500d, sp500m
run_dating_alg

Runs the dating algorithm to identify Bull and Bear states

Description
This function implements the dating algorithm of Bry and Boschan (1971) to identify Bull and Bear states.

Usage
run_dating_alg(index)

Arguments

index           vector containing the stock price index

Value
A logical vector that contains TRUE for Bull states and FALSE for Bear states.

Note
Be aware that the states in the beginning and in the end of "index" are not properly defined. The users are advised to always visually check the correctness of the result (during Bull states the prices should generally increase, during the Bear states decrease).

References

run_filtering_alg

Runs the filtering algorithm to identify Bull and Bear states

Description
This function implements the filtering algorithm of Lunde and Timmermann (2004) to identify Bull and Bear states.

Usage
run_filtering_alg(index)
**setpar_dating_alg**

**Arguments**

- `index` vector containing the stock price index

**Value**

A logical vector that contains TRUE for Bull states and FALSE for Bear states

**Note**

Be aware that the states in the beginning and in the end of "index" are not properly defined

**References**


---

**setpar_dating_alg**  
Sets the parameters of the dating algorithm

**Description**

This function sets the parameters of the dating algorithm of Bry and Boschan (1971)

**Usage**

```r
setpar_dating_alg(t_window, t_censor, t_phase, t_cycle, max_chng)
```

**Arguments**

- `t_window` the half-size of the rolling window to find minima and maxima
- `t_censor` the size of the left and right margin (for the censoring operation)
- `t_phase` the minimum phase (bull or bear) length
- `t_cycle` the minimum full cycle length
- `max_chng` the change (in percentages) in the "index" that invalidates the minimum phase length rule

**Value**

None

**Note**

All parameters but "max_chng" are given in a number of observations. For example, if data are at the monthly frequency, "t_cycle=16" defines that the minimum cycle length should be 16 months.
References


---

**setpar_filtering_alg**  _Sets the parameters of the filtering algorithm_

**Description**

This function sets the parameters of the filtering algorithm of Lunde and Timmermann (2004)

**Usage**

```r
setpar_filtering_alg(tr_bull, tr_bear)
```

**Arguments**

- `tr_bull` threshold to identify a Bull state (in percentages)
- `tr_bear` threshold to identify a Bear state (in percentages)

**Value**

None

**References**


---

**sp500d**  _Daily data the S&P 500 index_

**Description**

The daily closing prices of the S&P 500 index

**Usage**

```r
sp500d
```
sp500m

Format
An object of class "zoo" containing daily closing prices of the S&P 500 index starting from January 1950

Source
https://finance.yahoo.com/

See Also
Other data sets: djiad, djiam, sp500m

sp500m Monthly data on the S&P 500 index

Description
Monthly closing prices of the S&P 500 index

Usage
sp500m

Format
An object of class "zoo" containing monthly closing prices of the S&P 500 index starting from January 1950

Source
https://finance.yahoo.com/

See Also
Other data sets: djiad, djiam, sp500d
Index

* **data sets**
  
  - djiad, 4
  - djiam, 5
  - sp500d, 8
  - sp500m, 9

* **datasets**
  
  - djiad, 4
  - djiam, 5
  - sp500d, 8
  - sp500m, 9

  bb.dating.states, 2
  bb.plot, 3
  bb.summary.stat, 4

  djiad, 4, 5, 9
  djiam, 5, 5, 9

  run_dating_alg, 2–4, 6
  run_filtering_alg, 2–4, 6

  setpar_dating_alg, 7
  setpar_filtering_alg, 8
  sp500d, 5, 8, 9
  sp500m, 5, 9, 9