Package ‘fmlr’

March 5, 2019

Title Financial Machine Learning using R

Version 0.0.8.0

Author Larry Lei Hua [aut, cre],

Maintainer Larry Lei Hua <larry.lei.hua@gmail.com>

Description This is an R package for tools of machine learning for financial data.

Depends R (>= 3.5)

License file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

LinkingTo Rcpp

Imports stats,
lubridate,
pracma,
zoo,
Rcpp,
stringr,
readr

R topics documented:

`acc_lucky` *A function to check whether a classification is better than a guess*

Description

A function to check whether a classification is better than a guess

Usage

```
acc_lucky(train_class, test_class, my_acc, s = 1000)
```

Arguments

`train_class`	a vector for the distribution of classes in the training set
`test_class`	a vector for the distribution of classes in the test set
`my_acc`	a number between 0 and 1 for the classification accuracy to be evaluated
`s`	sample size of simulations used to check p-values

Author(s)

Larry Lei Hua

Examples

```
train_class <- c(1223,1322,1144)
test_class <- c(345,544,233)
my_acc <- 0.45
acc_lucky(train_class, test_class, my_acc)
```

bar_tick *Construct tick bars*

Description

Construct tick bars

Usage

```
bar_tick(dat, nTic)
```

Arguments

`dat`	dat input with at least the following columns: Price, Size
`nTic`	the number of ticks of each window

Value

time stamp at the end of each bar (if timestamp is provided), and H,L,O,C,V

Author(s)

Larry Lei Hua

bar_tick_imbalance *Construct tick imbalance bars*

Description

Construct tick imbalance bars

Usage

```
bar_tick_imbalance(dat, w0 = 10, bkw_T = 5, bkw_b = 5)
```

Arguments

`dat`	dat input with at least the following column: Price, Size
`w0`	the time window length of the first bar
`bkw_T`	backward window length when using pracma::movavg for exponentially weighted average T
`bkw_b`	backward window length when using pracma::movavg for exponentially weighted average b_t

Value

a list of vectors for tStamp (if returned), and HLOCV of tick imbalance bars. Note that the remaining data after the latest imbalance time point will be formed as a bar.

Author(s)

Larry Lei Hua

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 4), runif(50)), Size = rep(10,54))
bar_tick_imbalance(dat)
```

`bar_tick_runs` *Construct tick runs bars*

Description

Construct tick runs bars

Usage

```
bar_tick_runs(dat, w0 = 10, de = 1, bkw_T = 5, bkw_Pb1 = 5,
  filter = FALSE)
```

Arguments

`dat`	dat input with at least the following column: Price, Size
`w0`	the time window length of the first bar
`de`	a positive value for adjusting the expected window size, that is, de*E0T; default: 1
`bkw_T`	backward window length when using pracma::movavg for exponentially weighted average T
`bkw_Pb1`	backward window length when using pracma::movavg for exponentially weighted average P[b_t=1]
`filter`	whether used as a filter; default FALSE. If TRUE, then only i_feabar, the ending time index of feature bars, is returned

Value

If filter==FALSE, a list of vectors for tStamp (if returned), and HLOCV of tick runs bars. Note that the remaining data after the latest ending time point detected will be formed as a bar. If filter==TRUE, i_feabar a vector of integers for the time index.

Author(s)

Larry Lei Hua

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 4), runif(1000)), Size = rep(10,1004))
x1 <- bar_tick_runs(dat)
x2 <- bar_tick_runs(dat, filter=TRUE)
```

`bar_time` *Construct time bars*

Description

Construct time bars

Usage

```
bar_time(dat, tDur = 1)
```

Arguments

`dat`	dat input with at least the following columns: tStamp, Price, Size, where tStamp should be sorted already
`tDur`	the time duration in seconds of each window

Value

tStamp, seconds since the starting time point, and H,L,O,C,V

Author(s)

Larry Lei Hua

bar_unit *Construct unit bars*

Description

Construct unit bars

Usage

```
bar_unit(dat, unit)
```

Arguments

dat	dat input with at least the following columns: Price, Size. If timestamp is provided than output also contains timestamp of the unit bars
unit	the total dollar (unit) traded of each window

Value

time stamp at the end of each bar (if timestamp is provided), and H,L,O,C,V

Author(s)

Larry Lei Hua

bar_unit_runs *Construct unit runs bars*

Description

Construct unit runs bars

Usage

```
bar_unit_runs(dat, u_0 = 2000, w0 = 10, de = 1, bkw_T = 5,
  bkw_Pb1 = 5, bkw_U = 5, filter = FALSE)
```

Arguments

dat	dat input with at least the following column: Price, Size
u_0	average unit (volume*price) for each trade, and it is used to create the first bar
w0	the time window length of the first bar
de	a positive value for adjusting the expected window size, that is, de*E0T; default: 1

`bkw_T` backward window length when using pracma::movavg for exponentially weighted average T

`bkw_Pb1` backward window length when using pracma::movavg for exponentially weighted average P[b_t=1]

`bkw_U` backward window length for exponentially weighted average volumes

`filter` whether used as a filter; default FALSE. If TRUE, then only i_feabar, the ending time index of feature bars, is returned

Value

If filter==FALSE, a list of vectors for tStamp (if returned), and HLOCV of volume runs bars. Note that the remaining data after the latest ending time point detected will be formed as a bar. If filter==TRUE, i_feabar a vector of integers for the time index.

Author(s)

Larry Lei Hua

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 4), runif(50)), Size = floor(runif(54)*100))
bar_unit_runs(dat, u_0=mean(dat$Price)*mean(dat$Size))
bar_unit_runs(dat, u_0=mean(dat$Price)*mean(dat$Size), filter=TRUE)
```

`bar_volume` *Construct volume bars*

Description

Construct volume bars

Usage

```
bar_volume(dat, vol)
```

Arguments

`dat` dat input with at least the following columns: Price, Size

`vol` the volume of each window

Value

time stamp at the end of each bar (if timestamp is provided), and H,L,O,C,V

Author(s)

Larry Lei Hua

---

bar_volume_imbalance *Construct volume imbalance bars*

Description

Construct volume imbalance bars

Usage

```
bar_volume_imbalance(dat, w0 = 10, bkw_T = 5, bkw_b = 5)
```

Arguments

dat	dat input with at least the following column: Price, Size
w0	the time window length of the first bar
bkw_T	backward window length when using pracma::movavg for exponentially weighted average T
bkw_b	backward window length when using pracma::movavg for exponentially weighted average b_tv_t

Value

a list of vectors for tStamp (if returned), and HLOCV of volume imbalance bars. Note that the remaining data after the latest imbalance time point will be formed as a bar.

Author(s)

Larry Lei Hua

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 4), runif(50)), Size = rep(10,54))
bar_volume_imbalance(dat)
```

bar_volume_runs *Construct volume runs bars*

Description

Construct volume runs bars

Usage

```
bar_volume_runs(dat, v_0 = 20, w0 = 10, de = 1, bkw_T = 5,
  bkw_Pb1 = 5, bkw_V = 5, filter = FALSE)
```

Arguments

`dat`	dat input with at least the following column: Price, Size
`v_0`	average volume for each trade, and it is used to create the first bar
`w0`	the time window length of the first bar
`de`	a positive value for adjusting the expected window size, that is, de*E0T; default: 1
`bkw_T`	backward window length when using pracma::movavg for exponentially weighted average T
`bkw_Pb1`	backward window length when using pracma::movavg for exponentially weighted average P[b_t=1]
`bkw_V`	backward window length for exponentially weighted average volumes
`filter`	whether used as a filter; default FALSE. If TRUE, then only i_feabar, the ending time index of feature bars, is returned

Value

If filter==FALSE, a list of vectors for tStamp (if returned), and HLOCV of volume runs bars. Note that the remaining data after the latest ending time point detected will be formed as a bar. If filter==TRUE, i_feabar a vector of integers for the time index.

Author(s)

Larry Lei Hua

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 4), runif(50)), Size = floor(runif(54)*100))
bar_volume_runs(dat)
bar_volume_runs(dat, filter=TRUE)
```

`ema` *exponentially weighted moving average; only return the last value*

Description

exponentially weighted moving average; only return the last value

Usage

```
ema(x, n)
```

Arguments

`x`	a numeric vector
`n`	window size

Value

a numeric value

Author(s)

Larry Lei Hua

`fracDiff` *convert a time series into a fractionally differentiated series*

Description

convert a time series into a fractionally differentiated series

Usage

```
fracDiff(x, d = 0.5, nWei = 10, tau = NULL)
```

Arguments

`x`	a vector of time series to be fractionally differentiated
`d`	the order for fractionally differentiated features
`nWei`	number of weights for output
`tau`	threshold where weights are cut off; default is NULL, if not NULL then use tau and nWei is not used

Author(s)

Larry Lei Hua

`imbalance_tick` *The auxiliary function b_t for constructing tick imbalance bars. The first b_t is assigned the value 0 because no information is available*

Description

The auxiliary function b_t for constructing tick imbalance bars. The first b_t is assigned the value 0 because no information is available

Usage

```
imbalance_tick(dat)
```

Arguments

`dat`	dat input with at least the following columns: Price

Author(s)

Larry Lei Hua, ming08108(github)

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 4), runif(2), 0.5, 0.5, 0.4, runif(2) ))

b_t <- imbalance_tick(dat)
```

`imbalance_volume` *The auxiliary function b_tv_t for constructing volume imbalance bars. The first b_tv_t is assigned the value 0 because no information is available*

Description

The auxiliary function b_tv_t for constructing volume imbalance bars. The first b_tv_t is assigned the value 0 because no information is available

Usage

```
imbalance_volume(dat)
```

Arguments

`dat`	dat input with at least the following columns: Price, Size

Author(s)

Larry Lei Hua

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 4), runif(10)), Size = rep(10,14))

b_tv_t <- imbalance_volume(dat)
```

istar_CUSUM *time index that triggers a symmetric CUSUM filter*

Description

time index that triggers a symmetric CUSUM filter

Usage

```
istar_CUSUM(x, h)
```

Arguments

x	a vector of time series to be filtered
h	a vector of the thresholds

Author(s)

Larry Lei Hua

Examples

```
set.seed(1)
x <- runif(100, 1, 3)
h <- rep(1.5, 100)
i_CUSUM <- istar_CUSUM(x,h)
abline(v=i_CUSUM, lty = 2)

## Comparing C and R versions
# set.seed(1)
# x <- runif(1000000, 1, 3)
# h <- rep(1.5, 100)

# start_time <- Sys.time()
# i_CUSUM <- istar_CUSUM(x,h)
# end_time <- Sys.time()
# C_time <- end_time - start_time
# start_time <- Sys.time()
# i_CUSUM_R <- istar_CUSUM_R(x,h)
# end_time <- Sys.time()
# R_time <- end_time - start_time
# cat("C and R time: ", C_time, R_time)
# all(i_CUSUM-i_CUSUM_R==0)
```

istar_CUSUM_R

time index that triggers a symmetric CUSUM filter (R version for istar_CUSUM(), for shorter x, the R version can be faster than the C version)

Description

time index that triggers a symmetric CUSUM filter (R version for istar_CUSUM(), for shorter x, the R version can be faster than the C version)

Usage

```
istar_CUSUM_R(x, h)
```

Arguments

x	a vector of time series to be filtered
h	a vector of the thresholds

Author(s)

Larry Lei Hua

label_meta

Meta labeling, including three options: triple barriers, upper and vertical barriers, and lower and vertical barriers.

Description

Meta labeling, including three options: triple barriers, upper and vertical barriers, and lower and vertical barriers.

Usage

```
label_meta(x, events, ptSl, ex_vert = T, n_ex = 0)
```

Arguments

`x` a vector of prices series to be labeled.

`events` a dataframe that has the following columns:

- t0: event's starting time index.
- t1: event's ending time index; if t1==Inf then no vertical barrier, i.e., last observation in x is the vertical barrier.
- trgt: the unit absolute return used to set up the upper and lower barriers.
- side: 0: both upper and lower barriers; 1: only upper; -1: only lower.

`ptSl` a vector of two multipliers for the upper and lower barriers, respectively.

`ex_vert` whether exclude the output when the vertical barrier is hit; default is T.

`n_ex` number of excluded observations at the begining of x; default is 0.

Value

data frame with the following columns:

- T_up: local time index when the upper barrier is hit; Inf means that upper is not hit.
- T_lo: local time index when the lower barrier is hit; Inf means that lower is not hit.
- t1: local time index when the vertical barrier is hit.
- ret: return associated with the event.
- label: low:-1, vertical:0, upper:1.
- t0Fea: begining time index of feature bars.
- t1Fea: ending time index of feature bars.
- tLabel: ending time index of events, i.e., when the labels are created. Both t1Fea and tLabel will be useful for sequential bootstrap.

Author(s)

Larry Lei Hua

`purged_k_CV` *Purged k-fold CV with embargo*

Description

Purged k-fold CV with embargo

Usage

```
purged_k_CV(feaMat, k = 5, gam = 0.01)
```

Arguments

feaMat	a data.frame for feature matrix with the first column being the label
k	number of folds for k-fold CV
gam	gamma for embargo

Value

a list of k data.frame, each containing a test set and a training set

Author(s)

Larry Lei Hua

Examples

```
feaMat <- data.frame(Y = c(1,1,0,1,0),
                     V = c(2,4,2,4,1),
                     t1Fea = c(2,5,8,14,20),
                     tLabel = c(4,12,16,23,38))
purged_k_CV(feaMat, k=2, gam=0.1)
```

read_algoseek_futures_fullDepth
Load AlgoSeek Futures Full Depth data from zip files

Description

Load AlgoSeek Futures Full Depth data from zip files

Usage

```
read_algoseek_futures_fullDepth(zipdata, whichData = NULL)
```

Arguments

zipdata	the original zip data provided by AlgoSeek
whichData	the specific data to be loaded; by default load all data in the zip file

Author(s)

Larry Lei Hua

Examples

```
zipdata <- tempfile()
download.file("https://www.algoseek.com/static/files/sample_data/
futures_and_future_options/ESH5.Futures.FullDepth.20150128.csv.zip",zipdata)
dat <- read_algoseek_futures_fullDepth(zipdata)


# Do not run unless the file 20160104.zip is avaliable
# dat <- read_algoseek_futures_fullDepth("20160104.zip", whichData="ES/ESH6.csv")
```

Tstar_tib *Tstar index for Tick Imbalance Bars (bar_tib)*

Description

Tstar index for Tick Imbalance Bars (bar_tib)

Usage

```
Tstar_tib(dat, w0 = 10, bkw_T = 5, bkw_b = 5)
```

Arguments

`dat`	dat input with at least the following columns: Price
`w0`	the time window length of the first bar
`bkw_T`	backward window length when using pracma::movavg for exponentially weighted average T
`bkw_b`	backward window length when using pracma::movavg for exponentially weighted average b_t

Value

a vector for the lengths of the tick imbalance bars. For example, if the return is c(10,26), then the 2 tick imbalance bars are (0,10] and (10, 36]

Author(s)

Larry Lei Hua

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 4), runif(50)))
T_tib <- Tstar_tib(dat)
b_t <- imbalance_tick(dat)
cumsum(b_t)[cumsum(T_tib)] # check the accumulated b_t's where the imbalances occur
```

`Tstar_trb_cpp` *Tstar index for Tick Runs Bars (bar_trb)*

Description

Tstar index for Tick Runs Bars (bar_trb)

Usage

```
Tstar_trb_cpp(b_t, w0, de, bkw_T, bkw_Pb1)
```

Arguments

`b_t`	output of imbalance_tick(dat) with the dat has at least the following columns: Price
`w0`	the time window length of the first bar
`de`	a positive value for adjusting the expected window size, that is, de*E0
`bkw_T`	backward window length for exponentially weighted average T
`bkw_Pb1`	backward window length for exponentially weighted average P[b_t=1]

Value

a list of the following two vectors: a vector for the lengths of the tick imbalance bars. For example, if the return is c(10,26), then the 2 tick imbalance bars are (0,10] and (10, 36] a vector indicating up runs or down runs

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 5), runif(100)))
b_t <- imbalance_tick(dat)
T_trb <- Tstar_trb_cpp(b_t, 10, 1.0, 10, 10)
col <- ifelse(T_trb$Type==1, ”red”, ”blue”)
T <- cumsum(T_trb$Tstar)
plot(dat$Price)
for(i in 1:length(T)) abline(v = T[i], col = col[i])
```

Tstar_vib *Tstar index for Volume Imbalance Bars (bar_vib)*

Description

Tstar index for Volume Imbalance Bars (bar_vib)

Usage

```
Tstar_vib(dat, w0 = 10, bkw_T = 5, bkw_b = 5)
```

Arguments

dat	dat input with at least the following columns: Price, Size
w0	the time window length of the first bar
bkw_T	backward window length when using pracma::movavg for exponentially weighted average T
bkw_b	backward window length when using pracma::movavg for exponentially weighted average b_tv_t

Value

a vector for the lengths of the tick imbalance bars. For example, if the return is c(10,26), then the 2 tick imbalance bars are (0,10] and (10, 36]

Author(s)

Larry Lei Hua

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 4), runif(50)), Size = rep(10, 54))
T_vib <- Tstar_vib(dat)
b_tv_t <- imbalance_volume(dat)
cumsum(b_tv_t)[cumsum(T_vib)] # check the accumulated b_t's where the imbalances occur
```

Tstar_vrb_cpp *Tstar index for Volume Runs Bars (bar_vrb)*

Description

Tstar index for Volume Runs Bars (bar_vrb)

Usage

```
Tstar_vrb_cpp(b_t, v_t, v_0, w0, de, bkw_T, bkw_Pb1, bkw_V)
```

Arguments

b_t	output of imbalance_tick(dat) with the data 'dat' has at least the following columns: Price
v_t	volume of the same data
v_0	average volume for each trade, and it is used to create the first bar
w0	the time window length of the first bar
de	a positive value for adjusting the expected window size, that is, de*E0T; default: 1.
bkw_T	backward window length for exponentially weighted average T
bkw_Pb1	backward window length for exponentially weighted average P[b_t=1]
bkw_V	backward window length for exponentially weighted average volumes

Value

a list of the following two vectors: a vector for the lengths of the tick imbalance bars. For example, if the return is c(10,26), then the 2 tick imbalance bars are (0,10] and (10, 36] a vector indicating up runs or down runs

Examples

```
set.seed(1)
dat <- data.frame(Price = c(rep(0.5, 5), runif(100)), Size = runif(105, 10, 100))
b_t <- imbalance_tick(dat)
v_t <- dat$Size
T_vrb <- Tstar_vrb_cpp(b_t, v_t, 55, 10, 1.0, 10, 10, 10)
col <- ifelse(T_vrb$Type==1, "red", "blue")
T <- cumsum(T_vrb$Tstar)
plot(dat$Price)
for(i in 1:length(T)) abline(v = T[i], col = col[i])
```

`weights_fracDiff` *calculate the weights for deriving fractionally differentiated series*

Description

calculate the weights for deriving fractionally differentiated series

Usage

```
weights_fracDiff(d = 0.5, nWei = 10, tau = NULL)
```

Arguments

`d`	the order for fractionally differentiated features
`nWei`	number of weights for output
`tau`	threshold where weights are cut off; default is NULL, if not NULL then use tau and nWei is not used

Author(s)

Larry Lei Hua

Examples

```
weights_fracDiff(0.5,tau=1e-3)
```

Index