Package ‘JFE’

December 5, 2021

Type Package

Title Tools and GUI for Analyzing Time Series Data of Just Finance and Econometrics

Version 2.5.2

Date 2021-12-05

Author Ho Tsung-wu

Maintainer Ho Tsung-wu <tsungwu@ntnu.edu.tw>

Description Support the analysis of financial and econometric time series, including recursive forecasts for machine learning.

License GPL (>= 2)

LazyData TRUE

LazyLoad yes

Depends R (>= 3.5),xts,fPortfolio

Imports caret, magrittr, tcltk, tcltk2, zoo

Suggests BurStFin, data.table, DescTools, fAssets, fBasics, forecast, FRAPO, h2o, iClick, keras, MASS, quantmod, rugarch, tensorflow, tibble, timeDate, timeSeries, timetk,

NeedsCompilation no

Repository CRAN

Date/Publication 2021-12-05 06:00:02 UTC

R topics documented:

`ActivePremium` *Active Premium or Active Return*

Description

The return on an investment's annualized return minus the benchmark's annualized return.

Usage

```
ActivePremium(Ra, Rb, scale = NA, ...)
```

Arguments

Ra	return vector of the portfolio
Rb	return vector of the benchmark asset
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
...	any other passthru parameters to Return.annualized (e.g., `geometric=FALSE`)

Details

Active Premium = Investment's annualized return - Benchmark's annualized return. With a view to speeding computation. I re-write the code of some ratios of the package PerformanceAnalytics, and use the same name for comparing the performance enhancing. Interested readers may compare speed improvement with the use of system.time().

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*, Fall 1994, 49-58.
See aslo package `PerformanceAnalytics`.

See Also

`InformationRatio` `TrackingError` `Return.annualized`

Examples

```
  data(assetReturns)
assetReturns=assetReturns["2011::2018"] #short sample for fast example
Ra=assetReturns[, -29]
Rb=assetReturns[,29] #DJI
  ActivePremium(Ra, Rb)
```

`AdjustedSharpeRatio` *Adjusted Sharpe ratio of the return distribution*

Description

Adjusted Sharpe ratio was introduced by Pezier and White (2006) to adjusts for skewness and kurtosis by incorporating a penalty factor for negative skewness and excess kurtosis.

Usage

```
AdjustedSharpeRatio(R, Rf = 0, FUN = "StdDev",...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rf`	the risk free rate
`FUN`	one of "StdDev" or "VaR" or "ES" to use as the denominator for unadjusted Sharpe ratio, default="StdDev"
`...`	any other pass through parameters

Details

$$AdjustedSharpeRatio = SR * [1 + (\frac{S}{6}) * SR - (\frac{K-3}{24}) * SR^2]$$

where SR is the sharpe ratio with data annualized, S is the skewness and K is the kurtosis

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.99.

Pezier, Jaques and White, Anthony. 2006. The Relative Merits of Investable Hedge Fund Indices and of Funds of Hedge Funds in Optimal Passive Portfolios. Check `https://econpapers.repec.org/paper/rdgicmadp/icma-dp2006-10.htm`
See also package `PerformanceAnalytics`.

See Also

`SharpeRatio.annualized`

Examples

```
    data(assetReturns)

AdjustedSharpeRatio(assetReturns)
```

AppraisalRatio *Appraisal ratio of the return distribution*

Description

Appraisal ratio is the Jensen's alpha adjusted for specific risk. The numerator is divided by specific risk instead of total risk.

Usage

```
AppraisalRatio(Ra, Rb, Rf = 0, method = c("appraisal", "modified",
  "alternative"), ...)
```

Arguments

`Ra`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rb`	return vector of the benchmark asset
`Rf`	risk free rate, in same period as your returns
`method`	is one of "appraisal" to calculate appraisal ratio, "modified" to calculate modified Jensen's alpha or "alternative" to calculate alternative Jensen's alpha.
`...`	any other pass through parameters

Details

Modified Jensen's alpha is Jensen's alpha divided by beta.

Alternative Jensen's alpha is Jensen's alpha divided by systematic risk.

$$Appraisal ratio = \frac{\alpha}{\sigma_{\epsilon}}$$

$$Modified Jensen's alpha = \frac{\alpha}{\beta}$$

$$Alternative Jensen's alpha = \frac{\alpha}{\sigma_S}$$

where $alpha$ is the Jensen's alpha, $\sigma_{epsilon}$ is the specific risk, σ_S is the systematic risk.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.77.
See also package `PerformanceAnalytics`.

Examples

```
    data(assetReturns)
Ra=assetReturns[, -29]
Rb=assetReturns[,29] #DJI
AppraisalRatio(Ra, Rb, method="appraisal")
```

`BernardoLedoitRatio` *Bernardo and Ledoit ratio of the return distribution*

Description

To calculate Bernardo and Ledoit ratio we take the sum of the subset of returns that are above 0 and we divide it by the opposite of the sum of the subset of returns that are below 0

Usage

```
BernardoLedoitRatio(R, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`...`	any other passthru parameters

Details

$$BernardoLedoitRatio(R) = \frac{\frac{1}{n}\sum^{n}_{t=1} max(R_{t},0)}{\frac{1}{n}\sum^{n}_{t=1} max(-R_{t},0)}$$

where n is the number of observations of the entire series

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.95.
See also package `PerformanceAnalytics`.

Examples

```
    data(assetReturns)
BernardoLedoitRatio(R=assetReturns)
```

BurkeRatio *Burke ratio of the return distribution*

Description

To calculate Burke ratio we take the difference between the portfolio return and the risk free rate and we divide it by the square root of the sum of the square of the drawdowns. To calculate the modified Burke ratio we just multiply the Burke ratio by the square root of the number of datas.

Usage

```
BurkeRatio(R, Rf = 0, modified = FALSE, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rf	the risk free rate
modified	a boolean to decide which ratio to calculate between Burke ratio and modified Burke ratio.
...	any other passthru parameters

Details

$$BurkeRatio = \frac{r_P - r_F}{\sqrt{\sum^{d}_{t=1} {D_t}^2}}$$

$$ModifiedBurkeRatio = \frac{r_P - r_F}{\sqrt{\sum^{d}_{t=1} \frac{{D_t}^2}{n}}}$$

where n is the number of observations of the entire series, d is number of drawdowns, r_P is the portfolio return, r_F is the risk free rate and D_t the t^{th} drawdown.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.90-91.
See aslo package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
assetReturns=assetReturns["2011::2018"] #short sample for fast example
  BurkeRatio(assetReturns,Rf=0)
```

`CalmarRatio`	*calculate a Calmar or Sterling reward/risk ratio Calmar and Sterling Ratios are yet another method of creating a risk-adjusted measure for ranking investments similar to the* `SharpeRatio`*.*

Description

Both the Calmar and the Sterling ratio are the ratio of annualized return over the absolute value of the maximum drawdown of an investment. The Sterling ratio adds an excess risk measure to the maximum drawdown, traditionally and defaulting to 10%.

Usage

```
CalmarRatio(R, scale = NA)

SterlingRatio(R, scale = NA, excess = 0.1)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`scale`	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
`excess`	for Sterling Ratio, excess amount to add to the max drawdown, traditionally and default .1 (10%)

Details

It is also traditional to use a three year return series for these calculations, although the functions included here make no effort to determine the length of your series. If you want to use a subset of your series, you'll need to truncate or subset the input data to the desired length.

Many other measures have been proposed to do similar reward to risk ranking. It is the opinion of this author that newer measures such as Sortino's `SharpeRatio` are both "better" measures, and should be preferred to the Calmar or Sterling Ratio.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Bacon, Carl. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004.
See also package `PerformanceAnalytics`.

See Also

`Return.annualized`,
`maxDrawdown`,

Examples

```
  data(assetReturns)
R=assetReturns[, -29]

    SterlingRatio(R)
```

`CAPM.jensenAlpha` *Jensen's alpha of the return distribution*

Description

The Jensen's alpha is the intercept of the regression equation in the Capital Asset Pricing Model and is in effect the exess return adjusted for systematic risk.

Usage

```
CAPM.jensenAlpha(Ra, Rb, Rf = 0, ...)
```

Arguments

`Ra`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rb`	return vector of the benchmark asset
`Rf`	risk free rate, in same period as your returns
`...`	any other passthru parameters

Details

$$\alpha = r_p - r_f - \beta_p * (b - r_f)$$

where r_f is the risk free rate, β_r is the regression beta, r_p is the portfolio return and b is the benchmark return

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.72
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
assetReturns=assetReturns["2011::2018"] #short sample for fast example
Ra=assetReturns[, -29]
Rb=assetReturns[,29] #DJI

CAPM.jensenAlpha(Ra, Rb)
```

`data-sets` *Assets Data Sets*

Description

assetReturns contains DJ component stocks returns data. macrodata contains US unemployment(unrate) and year-to-year changes in three regional business cycle indices (OED, NAFTA, and G7).

Usage

```
data(assetReturns)
data(macrodata)
```

Value

assetReturns is a time series object of package `"xts"`; the others are time series objects of package `"timeSeries"`.

`downloadStockAI` *Download time series data from stock-ai.com*

Description

It downloads financial and economic time series data from the web stock-ai.com.

Usage

```
downloadStockAI(key="5edl69aag5",var.name="TWECO", from="2006-01-01", to="2015-12-31",
                                showdata=TRUE)
```

Arguments

`key`	key issued from the web, default is "5edl69aag5".
`var.name`	The name of variable to be downloaded, the default is Taiwan's economic growth "TWECO", one variable for each download. The default time range is ten years.
`from`	The start date of var.name, the default is "2006-01-01".
`to`	The end date of var.name, the default is "2015-12-31".
`showdata`	Whether the downloaded variable is shown on screen. The default is TRUE.

Details

This function connects with stock-ai.com and downloads the specified data.

Value

`y`	The timeSeries data object

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

Examples

```
# Cross-validation takes time, example below is commented.
#library(timeSeries)
#downloadStockAI(var.name="UNRATE") #US unemployment rate
#downloadStockAI() #Taiwan economic growth rate
```

DownsideDeviation *downside risk (deviation, variance) of the return distribution*

Description

Downside deviation, semideviation, and semivariance are measures of downside risk.

Usage

```
DownsideDeviation(R, MAR = 0, method = c("full", "subset"),potential = FALSE,...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`MAR`	Minimum Acceptable Return, in the same periodicity as your returns
`method`	one of "full" or "subset", indicating whether to use the length of the full series or the length of the subset of the series below the MAR as the denominator, defaults to "full"
`potential`	if TRUE, calculate downside potential instead, default FALSE
`...`	any other passthru parameters

Details

Downside deviation, similar to semi deviation, eliminates positive returns when calculating risk. Instead of using the mean return or zero, it uses the Minimum Acceptable Return as proposed by Sharpe (which may be the mean historical return or zero). It measures the variability of underperformance below a minimum targer rate. The downside variance is the square of the downside potential.

To calculate it, we take the subset of returns that are less than the target (or Minimum Acceptable Returns (MAR)) returns and take the differences of those to the target. We sum the squares and divide by the total number of returns to get a below-target semi-variance.

$$DownsideDeviation(R, MAR) = \delta_{MAR} = \sqrt{\sum_{t=1}^{n} \frac{min[(R_t - MAR), 0]^2}{n}}$$

$$DownsideVariance(R, MAR) = \sum_{t=1}^{n} \frac{min[(R_t - MAR), 0]^2}{n}$$

$$DownsidePotential(R, MAR) = \sum_{t=1}^{n} \frac{min[(R_t - MAR), 0]}{n}$$

where n is either the number of observations of the entire series or the number of observations in the subset of the series falling below the MAR.

SemiDeviation or SemiVariance is a popular alternative downside risk measure that may be used in place of standard deviation or variance. SemiDeviation and SemiVariance are implemented as a wrapper of DownsideDeviation with MAR=mean(R).

In many functions like Markowitz optimization, semideviation may be substituted directly, and the covariance matrix may be constructed from semideviation or the vector of returns below the mean rather than from variance or the full vector of returns.

In semideviation, by convention, the value of n is set to the full number of observations. In semi-variance the the value of n is set to the subset of returns below the mean. It should be noted that while this is the correct mathematical definition of semivariance, this result doesn't make any sense if you are also going to be using the time series of returns below the mean or below a MAR to construct a semi-covariance matrix for portfolio optimization.

Sortino recommends calculating downside deviation utilizing a continuous fitted distribution rather than the discrete distribution of observations. This would have significant utility, especially in cases of a small number of observations. He recommends using a lognormal distribution, or a fitted distribution based on a relevant style index, to construct the returns below the MAR to increase the confidence in the final result. Hopefully, in the future, we'll add a fitted option to this function, and would be happy to accept a contribution of this nature.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Sortino, F. and Price, L. Performance Measurement in a Downside Risk Framework. *Journal of Investing*. Fall 1994, 59-65.
Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008

Plantinga, A., van der Meer, R. and Sortino, F. The Impact of Downside Risk on Risk-Adjusted Performance of Mutual Funds in the Euronext Markets. July 19, 2001. Available at SSRN:`https://www.ssrn.com/abstract=277352`
see especially end note 10 `https://en.wikipedia.org/wiki/Semivariance`.

See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
assetReturns=assetReturns["2011::2018"] #short sample for fast example
R=assetReturns[, -29]
  DownsideDeviation(R, MAR = 0)
```

`DRatio` *d ratio of the return distribution*

Description

The d ratio is similar to the Bernado Ledoit ratio but inverted and taking into account the frequency of positive and negative returns.

Usage

```
DRatio(R, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`...`	any other passthru parameters

Details

It has values between zero and infinity. It can be used to rank the performance of portfolios. The lower the d ratio the better the performance, a value of zero indicating there are no returns less than zero and a value of infinity indicating there are no returns greater than zero.

$$DRatio(R) = \frac{n_d * \sum^{n}_{t=1} max(-R_t, 0)}{n_u * \sum^{n}_{t=1} max(R_t, 0)}$$

where n is the number of observations of the entire series, n_d is the number of observations less than zero, n_u is the number of observations greater than zero

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.95
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]

  DRatio(R)
```

DrawdownPeak — *Drawdawn peak of the return distribution*

Description

Drawdawn peak is for each return its drawdown since the previous peak

Usage

```
DrawdownPeak(R, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`...`	any other passthru parameters

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]
# Not run
# DrawdownPeak(R)
```

InformationRatio — *InformationRatio = ActivePremium/TrackingError*

Description

The Active Premium divided by the Tracking Error.

Usage

```
InformationRatio(Ra, Rb, scale = NA)
```

Arguments

`Ra`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rb`	return vector of the benchmark asset
`scale`	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)

Details

InformationRatio = ActivePremium/TrackingError

This relates the degree to which an investment has beaten the benchmark to the consistency with which the investment has beaten the benchmark.

Note

William Sharpe now recommends `InformationRatio` preferentially to the original `SharpeRatio`.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.
See also package `PerformanceAnalytics`.

See Also

`TrackingError`
`ActivePremium`
`SharpeRatio`

Examples

```
  data(assetReturns)
Ra=assetReturns[, -29]
Rb=assetReturns[,29] #DJI

  InformationRatio(Ra, Rb)
```

`JFE` *Display the JFE User Interface*

Description

Start the JFE GUI (graphical user interface)

Usage

```
JFE()
```

Details

After loading the package, in the command prompt, type JFE() to start it. JFE is a menu-driven GUI designed to support the analysis of financial time series data with the aid of several R packages. The version 1.1 focuses on: Firstly, price visualization, including technical charting(by package quantmod); secondly, assets selection based on Performance index(by package PerformanceAnalytics); thirdly, portfolio optimization (by package `"fPORTFOLIO"`).
This command is an internal function to start the JFE GUI. To avoid unexpected problems of time series object, the imported data must be time series object (xts, or timeSeries) loaded by either .RData or .rda, file of .csv or other format is not supported; that is to say, users have only to know how to construct a R time-series object.
If execution of All-in-one from backtesting fails, then it is a problem associated with undocumented functions. Please re-install this package from Github via devtools::install_github("tsungwu/JFE"), detailed are also explained in Github and http://web.ntnu.edu.tw/~tsungwu/R_DevOps/R_DevOps.htm.

Value

Generate a menu-driven GUI

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

Examples

```
JFE()
```

`KellyRatio` *calculate Kelly criterion ratio (leverage or bet size) for a strategy*

Description

Kelly criterion ratio (leverage or bet size) for a strategy.

Usage

```
KellyRatio(R, Rf = 0)
```

Arguments

`R`	a vector of returns to perform a mean over
`Rf`	risk free rate, in same period as your returns

Details

The Kelly Criterion was identified by Bell Labs scientist John Kelly, and applied to blackjack and stock strategy sizing by Ed Thorpe.

The Kelly ratio can be simply stated as: “bet size is the ratio of edge over odds.” Mathematically, you are maximizing log-utility. As such, the Kelly criterion is equal to the expected excess return of the strategy divided by the expected variance of the excess return, or

$$leverage = \frac{(\overline{R}_s - R_f)}{StdDev(R)^2}$$

As a performance metric, the Kelly Ratio is calculated retrospectively on a particular investment as a measure of the edge that investment has over the risk free rate. It may be use as a stack ranking method to compare investments in a manner similar to the various ratios related to the Sharpe ratio.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Thorp, Edward O. (1997; revised 1998). The Kelly Criterion in Blackjack, Sports Betting, and the Stock Market. https://en.wikipedia.org/wiki/Kelly_criterion
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]

  KellyRatio(R, Rf=0)
```

`M2Sortino` *M squared for Sortino of the return distribution*

Description

M squared for Sortino is a M^2 calculated for Downside risk instead of Total Risk

Usage

```
M2Sortino(Ra, Rb, MAR = 0, ...)
```

Arguments

`Ra`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset return
`Rb`	return vector of the benchmark asset
`MAR`	the minimum acceptable return
`...`	any other passthru parameters

Details

$$M^2_S = r_P + Sortinoratio * (\sigma_{DM} - \sigma_D)$$

where M^2_S is MSquared for Sortino, r_P is the annualised portfolio return, σ_{DM} is the benchmark annualised downside risk and D is the portfolio annualised downside risk

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.102-103
See aslo package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
Ra=assetReturns[, -29]
Rb=assetReturns[,29] #DJI

M2Sortino(Ra, Rb, MAR=0)
```

`MartinRatio` *Martin ratio of the return distribution*

Description

To calculate Martin ratio we divide the difference of the portfolio return and the risk free rate by the Ulcer index

Usage

```
MartinRatio(R, Rf = 0, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rf`	risk free rate, in same period as your returns
`...`	any other passthru parameters

Details

$$Martinratio = \frac{r_P - r_F}{\sqrt{\sum_{i=1}^{n} \frac{D_i'^2}{n}}}$$

where r_P is the annualized portfolio return, r_F is the risk free rate, n is the number of observations of the entire series, D_i' is the drawdown since previous peak in period i

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.91
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]

# Not run
# MartinRatio(R)
```

`maxDrawdown` *caclulate the maximum drawdown from peak equity*

Description

To find the maximum drawdown in a return series, we need to first calculate the cumulative returns and the maximum cumulative return to that point. Any time the cumulative returns dips below the maximum cumulative returns, it's a drawdown. Drawdowns are measured as a percentage of that maximum cumulative return, in effect, measured from peak equity.

Usage

```
maxDrawdown(R, geometric = TRUE, invert = TRUE, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`geometric`	utilize geometric chaining (TRUE) or simple/arithmetic chaining (FALSE) to aggregate returns, default TRUE
`invert`	TRUE/FALSE whether to invert the drawdown measure. see Details.
`...`	any other passthru parameters

Details

The option to `invert` the measure should appease both academics and practitioners. The default option `invert=TRUE` will provide the drawdown as a positive number. This should be useful for optimization (which usually seeks to minimize a value), and for tables (where having negative signs in front of every number may be considered clutter). Practitioners will argue that drawdowns denote losses, and should be internally consistent with the quantile (a negative number), for which `invert=FALSE` will provide the value they expect. Individually, different preferences may apply for clarity and compactness. As such, we provide the option, but make no value judgment on which approach is preferable.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 88
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]

  maxDrawdown(R)
```

`MeanAbsoluteDeviation` *Mean absolute deviation of the return distribution*

Description

To calculate Mean absolute deviation we take the sum of the absolute value of the difference between the returns and the mean of the returns and we divide it by the number of returns.

Usage

```
MeanAbsoluteDeviation(R, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
...	any other passthru parameters

Details

$$MeanAbsoluteDeviation = \frac{\sum^{n}_{i=1} \mid r_i - \overline{r} \mid}{n}$$

where n is the number of observations of the entire series, r_i is the return in month i and $\overline{r}$ is the mean return

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.62.
See also package PerformanceAnalytics.

Examples

```
  data(assetReturns)
assetReturns=assetReturns["2011::2018"] #short sample for fast example
R=assetReturns[, -29]

MeanAbsoluteDeviation(R)
```

OmegaSharpeRatio *Omega-Sharpe ratio of the return distribution*

Description

The Omega-Sharpe ratio is a conversion of the omega ratio to a ranking statistic in familiar form to the Sharpe ratio.

Usage

```
OmegaSharpeRatio(R, MAR = 0, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`MAR`	Minimum Acceptable Return, in the same periodicity as your returns
`...`	any other passthru parameters

Details

To calculate the Omega-Sharpe ration we subtract the target (or Minimum Acceptable Returns (MAR)) return from the portfolio return and we divide it by the opposite of the Downside Deviation.

$$OmegaSharpeRatio(R, MAR) = \frac{r_p - r_t}{\sum^n_{t=1} \frac{max(r_t - r_i, 0)}{n}}$$

where n is the number of observations of the entire series

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.95
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]
  OmegaSharpeRatio(R)
```

`PainIndex` *Pain index of the return distribution*

Description

The pain index is the mean value of the drawdowns over the entire analysis period. The measure is similar to the Ulcer index except that the drawdowns are not squared. Also, it's different than the average drawdown, in that the numerator is the total number of observations rather than the number of drawdowns.

Usage

```
PainIndex(R, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`...`	any other passthru parameters

Details

Visually, the pain index is the area of the region that is enclosed by the horizontal line at zero percent and the drawdown line in the Drawdown chart.

$$Painindex = \sum_{i=1}^{n} \frac{\mid D'_i \mid}{n}$$

where n is the number of observations of the entire series, D'_i is the drawdown since previous peak in period i

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.89, Becker, Thomas (2006) Zephyr Associates
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]
# Not run
# PainIndex(R)
```

`PainRatio` *Pain ratio of the return distribution*

Description

To calculate Pain ratio we divide the difference of the portfolio return and the risk free rate by the Pain index

Usage

```
PainRatio(R, Rf = 0, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rf`	risk free rate, in same period as your returns
`...`	any other passthru parameters

Details

$$Painratio = \frac{r_P - r_F}{\sum_{i=1}^{n} \frac{|D'_i|}{n}}$$

where r_P is the annualized portfolio return, r_F is the risk free rate, n is the number of observations of the entire series, D'_i is the drawdown since previous peak in period i

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.91
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
assetReturns=assetReturns["2011::2018"] #short sample for fast example
R=assetReturns[, -29]
  PainRatio(R)
```

`ProspectRatio` *Prospect ratio of the return distribution*

Description

Prospect ratio is a ratio used to penalise loss since most people feel loss greater than gain

Usage

```
ProspectRatio(R, MAR, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`MAR`	the minimum acceptable return
`...`	any other passthru parameters

Details

$$ProspectRatio(R) = \frac{\frac{1}{n} * \sum_{i=1}^{n}(Max(r_i, 0) + 2.25 * Min(r_i, 0) - MAR)}{\sigma_D}$$

where n is the number of observations of the entire series, MAR is the minimum acceptable return and σ_D is the downside risk

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.100
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]

  ProspectRatio(R, MAR=0)
```

`Return.annualized` *calculate an annualized return for comparing instruments with different length history*

Description

An average annualized return is convenient for comparing returns.

Usage

```
Return.annualized(R, scale = NA, geometric = TRUE)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`scale`	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
`geometric`	utilize geometric chaining (TRUE) or simple/arithmetic chaining (FALSE) to aggregate returns, default TRUE

Details

Annualized returns are useful for comparing two assets. To do so, you must scale your observations to an annual scale by raising the compound return to the number of periods in a year, and taking the root to the number of total observations:

$$prod(1 + R_a)^{\frac{scale}{n}} - 1 = \sqrt[n]{prod(1 + R_a)^{scale}} - 1$$

where scale is the number of periods in a year, and n is the total number of periods for which you have observations.

For simple returns (geometric=FALSE), the formula is:

$$\overline{R_a} \cdot scale$$

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Bacon, Carl. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 6
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]

  Return.annualized(R)
```

`riskOptimalPortfolio` *Compute risk optimal portfolios maxDD, aveDD and CDaR*

Description

It calls `FRAPO` to compute risk optimal portfolio satisfying the constraint of draw downs and returns a S4 object of class `fPORTFOLIO`.

Usage

```
riskOptimalPortfolio(data, Type="AveDD",value)
```

Arguments

`data`	timeSeries object of price data. Please remember the asset data must be price, not returns.
`Type`	Drawdown types, we call package FRAPO to support three methods:"maxDD","aveDD",and "CDaR". For details, please see document of package FRAPO.
`value`	Positive numerical number for Type.

Details

The risk optimal portfolio calls `FRAPO` and wrapp the results as a S4 object of class `fPORTFOLIO`, all get functions of `fPORTFOLIO` are applicable.

Value

returns an S4 object of class `"fPORTFOLIO"`.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Roncalli Thierry, *Introduction to Risk parity and Budgeting*, 2014, CRC inc..
See also packages `fPORTFOLIO` and `FRAPO`

Examples

```
# Risk optimal portfolio takes time, example below is commented.
#data(LPP2005,package="fPortfolio")
Data =  fPortfolio::LPP2005[,1:6]#select 6 assets price
Data.RET=timeSeries::returns(Data) # Transform into returns to compute VALUE below
#VALUE=abs(mean(drawdowns(apply(Data.RET,1,mean))))
#output=riskOptimalPortfolio(Data,Type="AveDD",value=VALUE) # data input must be price.
#show(output)
#getWeights(output)
#getCovRiskBudgets(output)
```

`riskParityPortfolio` *Compute risk parity portfolio*

Description

It calls `FRAPO` to compute portfolio weights with equal risk contribution, or equal covariance risk budget, then returns a S4 object of class `fPORTFOLIO`.

Usage

```
riskParityPortfolio(data, covmat="cov", strategy="minrisk",Type="MV")
```

Arguments

`data`	timeSeries object of returns data
`covmat`	Function to compute mltvariate covariance matrix, we support five methods:"cov","ledoitWolf","shrink"," The default is sample covariance "cov".
`strategy`	strategyPortfolio as in package fPortfolio, we support 5 cases in fPortfolio package: "GMVP","maxreturn","minrisk", "tangency" and "All Assets". The default is "minrisk".
`Type`	portfolio type as in package fPortfolio, the default is "MV".

Details

The risk parity portfolio has two options: the first is to select a subset of assets and compute risk parity weights. To this end, we implement one of four portfolio strategies: "GMVP","maxreturn","minrisk", "tangency". The idea is that each portfolio strategy will pick the desirable assets by assigning weights, the assets with non-zero weights are selected ones; afterwards, we compute risk parity weights of these assets. Secondly, for "All Assets", all assets are included and compute an optimal weight vector satisfying risk parity condition,namely, equal risk contribution or covariance risk budget.

GMVP or Global minimum risk Portfolio: The function `minvariancePortfolio` returns the portfolio with the minimal risk on the efficient frontier. To find the minimal risk point the target risk returned by the function `efficientPortfolio` is minimized.

tangency or maximal returns/risk ratio Portfolio: The function `tangencyPortfolio` returns the portfolio with the highest return/risk ratio on the efficient frontier. For the Markowitz portfolio this is the same as the Sharpe ratio. To find this point on the frontier the return/risk ratio calculated from the target return and target risk returned by the function .

minrisk or Minumum Risk: The function `minriskPortfolio` is an efficient portfolio which lies on the efficient frontier. The `efficientPortfolio` function returns the properties of the efficient portfolio as an S4 object of class `fPORTFOLIO`

maxreturn or Maximum Return Portfolio: The function `maxreturnPortfolio` returns the portfolio with the maximal return for a fixed target risk.

Risk parity portfolio calls `FRAPO`, which requires symmetric covariance matrices, so far we support only five covariance methods.

Value

returns an S4 object of class `"fPORTFOLIO"`.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Roncalli Thierry, *Introduction to Risk parity and Budgeting*, 2014, CRC inc..
See also packages `fPORTFOLIO` and `FRAPO`

Examples

```
data(assetReturns)
assetReturns=assetReturns[,11:15]
output=riskParityPortfolio(assetReturns, covmat="cov", strategy="minrisk")
show(output)
getWeights(output)
getCovRiskBudgets(output)
```

`SharpeRatio` *calculate a traditional or modified Sharpe Ratio of Return over StdDev or VaR or ES*

Description

The Sharpe ratio is simply the return per unit of risk (represented by variability). In the classic case, the unit of risk is the standard deviation of the returns.

Usage

```
SharpeRatio(R, Rf = 0, alpha = 0.05, FUN="StdDev",annualize=FALSE, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rf`	risk free rate, in same period as your returns
`alpha`	Tail probability for VaR or ES, default alpha=.05
`FUN`	one of "StdDev" or "VaR" or "ES" to use as the denominator
`annualize`	if TRUE, annualize the measure, default FALSE
`...`	any other passthru parameters to the VaR or ES functions

Details

$$\frac{\overline{(R_a - R_f)}}{\sqrt{\sigma_{(R_a - R_f)}}}$$

William Sharpe now recommends `InformationRatio` preferentially to the original Sharpe Ratio.

The higher the Sharpe ratio, the better the combined performance of "risk" and return.

As noted, the traditional Sharpe Ratio is a risk-adjusted measure of return that uses standard deviation to represent risk.

A number of papers now recommend using a "modified Sharpe" ratio using a Modified Cornish-Fisher VaR or CVaR/Expected Shortfall as the measure of Risk.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.

Laurent Favre and Jose-Antonio Galeano. Mean-Modified Value-at-Risk Optimization with Hedge Funds. Journal of Alternative Investment, Fall 2002, v 5.
See also package PerformanceAnalytics.

See Also

`SharpeRatio.annualized`
`InformationRatio`
`TrackingError`
`ActivePremium`
`SortinoRatio`

Examples

```
  data(assetReturns)
R=assetReturns[, -29]

SharpeRatio(R)
```

SharpeRatio.annualized

calculate annualized Sharpe Ratio

Description

The Sharpe Ratio is a risk-adjusted measure of return that uses standard deviation to represent risk.

Usage

```
SharpeRatio.annualized(R, Rf = 0, alpha=0.05,scale = NA, geometric = TRUE, FUN = "StdDev")
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rf`	risk free rate, in same period as your returns
`alpha`	Tail probability for VaR or ES, default alpha=.05
`scale`	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
`geometric`	utilize geometric chaining (TRUE) or simple/arithmetic chaining (FALSE) to aggregate returns,default TRUE
`FUN`	one of "StdDev" or "VaR" or "ES" to use as the denominator, default="StdDev"

Details

The Sharpe ratio is simply the return per unit of risk (represented by variance). The higher the Sharpe ratio, the better the combined performance of "risk" and return.

This function annualizes the number based on the scale parameter.

$$\frac{\sqrt[n]{prod(1+R_a)^{scale}}-1}{\sqrt{scale}\cdot\sqrt{\sigma}}$$

Using an annualized Sharpe Ratio is useful for comparison of multiple return streams. The annualized Sharpe ratio is computed by dividing the annualized mean monthly excess return by the annualized monthly standard deviation of excess return.

William Sharpe now recommends Information Ratio preferentially to the original Sharpe Ratio.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.
See also package `PerformanceAnalytics`.

See Also

`SharpeRatio`
`InformationRatio`
`TrackingError`
`ActivePremium`
`SortinoRatio`

Examples

```
  data(assetReturns)
R=assetReturns[, -29]
  SharpeRatio.annualized(R)
```

`SkewnessKurtosisRatio` *Skewness-Kurtosis ratio of the return distribution*

Description

Skewness-Kurtosis ratio is the division of Skewness by Kurtosis.

Usage

```
SkewnessKurtosisRatio(R, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`...`	any other pass through parameters

Details

It is used in conjunction with the Sharpe ratio to rank portfolios. The higher the rate the better.

$$SkewnessKurtosisRatio(R, MAR) = \frac{S}{K}$$

where S is the skewness and K is the Kurtosis

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.100
See also package `PerformanceAnalytics`.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]
  SkewnessKurtosisRatio(R)
```

SortinoRatio *calculate Sortino Ratio of performance over downside risk*

Description

Sortino proposed an improvement on the Sharpe Ratio to better account for skill and excess performance by using only downside semivariance as the measure of risk.

Usage

```
SortinoRatio(R, MAR = 0,...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
MAR	Minimum Acceptable Return, in the same periodicity as your returns
...	any other passthru parameters

Details

Sortino contends that risk should be measured in terms of not meeting the investment goal. This gives rise to the notion of "Minimum Acceptable Return" or MAR. All of Sortino's proposed measures include the MAR, and are more sensitive to downside or extreme risks than measures that use volatility(standard deviation of returns) as the measure of risk.

Choosing the MAR carefully is very important, especially when comparing disparate investment choices. If the MAR is too low, it will not adequately capture the risks that concern the investor, and if the MAR is too high, it will unfavorably portray what may otherwise be a sound investment. When comparing multiple investments, some papers recommend using the risk free rate as the MAR. Practitioners may wish to choose one MAR for consistency, several standardized MAR values for reporting a range of scenarios, or a MAR customized to the objective of the investor.

$$SortinoRatio = \frac{(\overline{R_a - MAR})}{\delta_{MAR}}$$

where δ_{MAR} is the `DownsideDeviation`.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Sortino, F. and Price, L. Performance Measurement in a Downside Risk Framework. *Journal of Investing*. Fall 1994, 59-65.
See also package `PerformanceAnalytics`.

See Also

SharpeRatio
DownsideDeviation
InformationRatio

Examples

```
  data(assetReturns)
R=assetReturns[, -29]

  SortinoRatio(R)
```

`table.AnnualizedReturns`
Annualized Returns Summary: Statistics and Stylized Facts

Description

Table of Annualized Return, Annualized Std Dev, and Annualized Sharpe

Usage

```
table.AnnualizedReturns(R, scale = NA, Rf = 0, geometric = TRUE,
  digits = 4)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
Rf	risk free rate, in same period as your returns
geometric	utilize geometric chaining (TRUE) or simple/arithmetic chaining (FALSE) to aggregate returns, default TRUE
digits	number of digits to round results to

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

See Also

Return.annualized
SharpeRatio.annualized

Examples

```
  data(assetReturns)
Ra=assetReturns[, -29]
  table.AnnualizedReturns(R=Ra)
```

TrackingError *Calculate Tracking Error of returns against a benchmark*

Description

A measure of the unexplained portion of performance relative to a benchmark.

Usage

```
TrackingError(Ra, Rb, scale = NA)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)

Details

Tracking error is calculated by taking the square root of the average of the squared deviations between the investment's returns and the benchmark's returns, then multiplying the result by the square root of the scale of the returns.

$$TrackingError = \sqrt{\sum \frac{(R_a - R_b)^2}{len(R_a)\sqrt{scale}}}$$

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.
See also package `PerformanceAnalytics`.

See Also

`InformationRatio` `TrackingError`

Examples

```
  data(assetReturns)
assetReturns=assetReturns["2011::2018"] #short sample for fast example
Ra=assetReturns[, -29]
Rb=assetReturns[,29] #DJI

  TrackingError(Ra, Rb)
```

`TreynorRatio` *calculate Treynor Ratio or modified Treynor Ratio of excess return over CAPM beta*

Description

The Treynor ratio is similar to the Sharpe Ratio, except it uses beta as the volatility measure (to divide the investment's excess return over the beta).

Usage

```
TreynorRatio(Ra, Rb, Rf = 0, scale = NA, modified = FALSE)
```

Arguments

`Ra`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rb`	return vector of the benchmark asset
`Rf`	risk free rate, in same period as your returns
`scale`	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
`modified`	a boolean to decide whether to return the Treynor ratio or Modified Treynor ratio

Details

To calculate modified Treynor ratio, we divide the numerator by the systematic risk instead of the beta.

Equation:

$$TreynorRatio = \frac{\overline{(R_a - R_f)}}{\beta_{a,b}}$$

$$ModifiedTreynorRatio = \frac{r_p - r_f}{\sigma_s}$$

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

https://en.wikipedia.org/wiki/Treynor_ratio, Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.77
See also package `PerformanceAnalytics`.

See Also

`SharpeRatio` `SortinoRatio`

Examples

```
  data(assetReturns)
assetReturns=assetReturns["2011::2018"] #short sample for fast example
Ra=assetReturns[, -29]
Rb=assetReturns[,29] #DJI

  TreynorRatio(Ra, Rb)
```

`ttsAutoML` *Train time series by automatic machine learning of* `h2o` *provided by H2O.ai*

Description

It generates both the static and recursive time series plots of H2O.ai object generated by package `h2o` provided by H2O.ai.

Usage

```
ttsAutoML(y,x=NULL,train.end,arOrder=2,xregOrder=0,maxSecs=30)
```

Arguments

`y`	The time series of target variable, or the dependent variable, with `timeSeries` format.
`x`	The time series of input variables, or the independent variables, with `timeSeries` format.
`train.end`	The end date of training data, must be specificed. The default dates of train.start and test.end are the start and the end of input data; and the test.start is the 1-period next of train.end.
`arOrder`	The autoregressive order of the target variable, which may be sequentially specifed like arOrder=1:5; or discontinuous lags like arOrder=c(1,3,5); zero is not allowed.

`xregOrder`	The distributed lag structure of the input variables, which may be sequentially specifed like xregOrder=1:5; or discontinuous lags like xregOrder=c(0,3,5); zero is allowed since contemporaneous correlation is allowed.
`maxSecs`	The maximal run time specified, in seconds. Default=20.

Details

This function calls the h2o.automl function from package `h2o` to execute automatic machine learning estimation. When execution finished, it computes two types of time series forecasts: static and recursive. The procedure of h2o.automl automatically generates a lot of time features.

Value

`staticData`	Two-column data with actual target series and the static forecasting series
`recursiveData`	Two-column data with actual target series and the recursive forecasting series
`staticF`	The static forecasting series of the period of test data
`recursiveF`	The recursive forecasting series of the period of test data
`static.Accuracy`	The accuracy measures of `forecast` of static forecasting series
`recursive.Accuracy`	The accuracy measures of `forecast` of recursive forecasting series
`method`	The train_model_list used in ttsAutoML
`data`	The data used by arOrder and xregOrder

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

Examples

```
# Cross-validation takes time, example below is commented.
#library(timeSeries)
data(macrodata)
Y <- macrodata[,"unrate"]
X <- macrodata[,-1]
train.end <- "2008-12-01"

#results<-ttsAutoML(y=Y, x=X, train.end,arOrder=c(2,3), xregOrder=c(1,2),maxSecs=300)
#head(results$data)
#results$static.Accuracy
#results$recursive.Accuracy
```

ttsCaret *Train time series by `caret` and produce two types of time series forecasts: static and recursive*

Description

It generates both the static and recursive time series plots of machine learning prediction object generated by package `caret`.

Usage

```
ttsCaret(y,x=NULL,method,train.end, arOrder=2,xregOrder=0,type,tuneLength =10)
```

Arguments

`y`	The time series of target variable, or the dependent variable, with `timeSeries` format.
`x`	The time series of input variables, or the independent variables, with `timeSeries` format.
`method`	The train_model_list of `caret`. While using this, make sure that the method allows regression. Methods in c("svm","rf","rpart","blasso","bridge","enet","gbm") are feasible.
`train.end`	The end date of training data, must be specificed.The default dates of train.start and test.end are the start and the end of input data; and the test.start is the 1-period next of train.end.
`arOrder`	The autoregressive order of the target variable, which may be sequentially specifed like arOrder=1:5; or discontinuous lags like arOrder=c(1,3,5); zero is not allowed.
`xregOrder`	The distributed lag structure of the input variables, which may be sequentially specifed like xregOrder=0:5; or discontinuous lags like xregOrder=c(0,3,5); zero is allowed since contemporaneous correlation is allowed.
`type`	The additional input variables. We have four selection: "none"=no other variables, "trend"=inclusion of time dummy, "season"=inclusion of seasonal dummies, "both"=inclusion of both trend and season. No default.
`tuneLength`	The same as the length specified in train function of package `caret`.

Details

This function calls the train function of package `caret` to execute estimation. When execution finished, we compute two types of time series forecasts: static and recursive.

Value

`staticData`	Two-column data with actual target series and the static forecasting series
`recursiveData`	Two-column data with actual target series and the recursive forecasting series
`staticF`	The static forecasting series of the period of test data
`recursiveF`	The recursive forecasting series of the period of test data
`static.Accuracy`	The accuracy measures of `forecast` of static forecasting series
`recursive.Accuracy`	The accuracy measures of `forecast` of recursive forecasting series
`method`	The train_model_list used in ttsCaret
`data`	The data used by arOrder, xregOrder, and type

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

Examples

```
# Cross-validation takes time, example below is commented.
#library(timeSeries)
data(macrodata)
Y <- macrodata[,"unrate"]
X <- macrodata[,-1]
train.end <- "2008-12-01"
#results <- ttsCaret(y=Y, x=X, method="svm", train.end,arOrder=c(2,3),
#          xregOrder=c(1,2), type="none",tuneLength =10)
#head(results$data)
#results$static.Accuracy
#results$recursive.Accuracy
```

`ttsDS` *Generates the data structure that is used for training(estimation) and validation*

Description

This function generates the dataset's lag structure that is used for the estimation of ttsCaret, ttsAutoML and ttsLSTM.

Usage

```
ttsDS(y,x=NULL, arOrder=2,xregOrder=0,type=NULL)
```

Arguments

`y`	The time series of target variable, or the dependent variable, with `timeSeries` format.
`x`	The time series of input variables, or the independent variables, with `timeSeries` format.
`arOrder`	The autoregressive order of the target variable, which may be sequentially specifed like arOrder=1:5; or discontinuous lags like arOrder=c(1,3,5); zero is not allowed.
`xregOrder`	The distributed lag structure of the input variables, which may be sequentially specifed like xregOrder=0:5; or discontinuous lags like xregOrder=c(0,3,5); zero is allowed since contemporaneous correlation is allowed.
`type`	The additional input variables. We have four selection: "none"=no other variables, "trend"=inclusion of time dummy, "season"=inclusion of seasonal dummies, "both"=inclusion of both trend and season. No default.

Details

This function generates the dataset's lag structure that is used for estimating ttsCaret, ttsAutoML, and ttsLSTM functions.

Value

`DF`	The data used by arOrder, xregOrder, and type

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

Examples

```
#library(timeSeries)
data(macrodata)
Y <- macrodata[,"unrate"]
X <- macrodata[,-1]
head(ttsDS(y=Y, x=X, arOrder=c(1:2),xregOrder=c(0,1), type="both"))
```

`ttsLSTM` *Train time series by LSTM of* `tensorflow` *provided by* `kera`

Description

It generates both the static and recursive time series plots of deep learning LSTM object generated by package `tensorflow` provided by `kera`.

Usage

```
ttsLSTM(y,x=NULL,train.end,arOrder=2,xregOrder=0,type,memoryLoops=10)
```

Arguments

`y`	The time series of target variable, or the dependent variable, with `timeSeries` format.
`x`	The time series of input variables, or the independent variables, with `timeSeries` format.
`train.end`	The end date of training data, must be specificed.The default dates of train.start and test.end are the start and the end of input data; and the test.start is the 1-period next of train.end.
`arOrder`	The autoregressive order of the target variable, which may be sequentially specifed like arOrder=1:5; or discontinuous lags like arOrder=c(1,3,5); zero is not allowed.Default is 2.
`xregOrder`	The distributed lag structure of the input variables, which may be sequentially specifed like xregOrder=1:5; or discontinuous lags like xregOrder=c(0,3,5); zero is allowed since contemporaneous correlation is allowed.
`type`	The additional input variables. We have four selection: "none"=no other variables, "trend"=inclusion of time dummy, "season"=inclusion of seasonal dummies, "both"=inclusion of both trend and season. No default.
`memoryLoops`	Length of LSTM learning network loop, to achieve better learning results, this not is suggested to be the same as the length of data row. Default is 10..

Details

This function calls the function fit of package `tensorflow` to execute Long-Short Term Memory (LSTM) estimation. When execution finished, it computes two types of time series forecasts: static and recursive.

Value

`staticData`	Two-column data with actual target series and the static forecasting series
`recursiveData`	Two-column data with actual target series and the recursive forecasting series
`staticF`	The static forecasting series of the period of test data
`recursiveF`	The recursive forecasting series of the period of test data
`static.Accuracy`	The accuracy measures of `forecast` of static forecasting series
`recursive.Accuracy`	The accuracy measures of `forecast` of recursive forecasting series
`method`	The train_model_list used in ttsLSTM
`data`	The data used by arOrder, xregOrder, and type

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

Examples

```
# Cross-validation takes time, example below is commented.
data(macrodata)
#Y <- macrodata[,"unrate"]
#X <- macrodata[,-1]
#train.end <- "2008-12-01"
#results <- ttsLSTM(y=Y, x=X, train.end, arOrder=c(2,3), xregOrder=c(1,2), type="none",
#          memoryLoops =10)
#head(results$data)
#results$static.Accuracy
#results$recursive.Accuracy
```

`ttsPlot` *Plot time series prediction performance*

Description

It generates both the static and recursive time series plots of machine learning prediction object generated by package `caret`.

Usage

```
ttsPlot(object,
which,
vertical,
main=NULL,
xlab="Time",
ylab="Value",
col1="black",
col2="red",
type="o",
ylim=NULL)
```

Arguments

`object`	Object of tts object.
`which`	If which="static", it plots the static prediction of insample model fit; if which="recursive", it plots the recursive prediction of insample model fit
`vertical`	The number of veritcal time line of seperating training and testing regimes.
`main`	Text appeared as main in R plot.
`xlab`	Text appeared as main in R plot, default is "time".
`ylab`	Text appeared as main in R plot, default is "Value".
`col1`	Line color appeared as main in R plot, default is "black".
`col2`	Prediction line color appeared as main in R plot, default is "red".
`type`	Line type appeared as main in R plot, default is "o.
`ylim`	ylim appeared as main in R plot, default is range of object data value.

Details

This function handles object generated by ttsCaret and plots two time series data: the actual series and model's forecast.

Value

Returns a time series plot.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

Examples

```
# Cross-validation takes time, example below is commented.
#library(timeSeries)
data(macrodata)
Y <- macrodata[,"unrate"]
X <- macrodata[,-1]
train.end <- "2008-12-01"
#results  <- ttsCaret(y=Y, x=X, method="svm", train.end,arOrder=c(2,3),
#          xregOrder=c(1,2), type="none",tuneLength =10)
#ttsPlot(results,which=c("static","recursive")[2],vertical=t0)
```

`UlcerIndex` *calculate the Ulcer Index*

Description

Developed by Peter G. Martin in 1987 (Martin and McCann, 1987) and named for the worry caused to the portfolio manager or investor. This is similar to drawdown deviation except that the impact of the duration of drawdowns is incorporated by selecting the negative return for each period below the previous peak or high water mark. The impact of long, deep drawdowns will have significant impact because the underperformance since the last peak is squared.

Usage

```
UlcerIndex(R, ...)
```

Arguments

`R`	a vector, matrix, data frame, timeSeries or zoo object of asset returns
`...`	any other passthru parameters

Details

UI = sqrt(sum[i=1,2,...,n](D'_i^2/n)) where D'_i = drawdown since previous peak in period i

DETAILS: This approach is sensitive to the frequency of the time periods involved and penalizes managers that take time to recover to previous highs.

REFERENCES: Martin, P. and McCann, B. (1989) The investor's Guide to Fidelity Funds: Winning Strategies for Mutual Fund Investors. John Wiley & Sons, Inc. Peter Martin's web page on UI: "http://www.tangotools.com/ui/ui.htm "

Test against spreadsheet at: "http://www.tangotools.com/ui/UlcerIndex.xls "
See also package `PerformanceAnalytics`.

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]
  maxDrawdown(R)
```

`VolatilitySkewness` *Volatility and variability of the return distribution*

Description

Volatility skewness is a similar measure to omega but using the second partial moment. It's the ratio of the upside variance compared to the downside variance. Variability skewness is the ratio of the upside risk compared to the downside risk.

Usage

```
VolatilitySkewness(R, MAR = 0, stat = c("volatility", "variability"), ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`MAR`	Minimum Acceptable Return, in the same periodicity as your returns
`stat`	one of "volatility", "variability" indicating whether to return the volatility skewness or the variability skweness
`...`	any other passthru parameters

Details

$$VolatilitySkewness(R, MAR) = \frac{\sigma_U^2}{\sigma_D^2}$$

$$VariabilitySkewness(R, MAR) = \frac{\sigma_U}{\sigma_D}$$

where σ_U is the Upside risk and σ_D is the Downside Risk

Author(s)

Ho Tsung-wu <tsungwu@ntnu.edu.tw>, College of Management, National Taiwan Normal University.

References

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008 p.97-98.
See also package PerformanceAnalytics.

Examples

```
  data(assetReturns)
R=assetReturns[, -29]
  VolatilitySkewness(R, MAR=0, stat="volatility")
```

Index