Package 'dispositionEffect'

May 30, 2022

Type Package

Title Analysis of Disposition Effect on Financial Portfolios

Version 1.0.1

Description Evaluate the presence of disposition effect and others irrational investor's behaviors based solely on investor's transactions and financial market data. Experimental data can also be used to perform the analysis. Four different methodologies are implemented to account for the different nature of human behaviors on financial markets.
Novel analyses such as portfolio driven and time series disposition effect are also allowed.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports dplyr, purrr, lubridate, magrittr, progress

Suggests devtools, knitr, rmarkdown, roxygen2, testthat, covr, tidyr, skimr, ggplot2, ggridges, furrr, future, foreach, doParallel, parallel, bench

RoxygenNote 7.1.2

VignetteBuilder knitr

URL `https://marcozanotti.github.io/dispositionEffect/`, `https://github.com/marcozanotti/dispositionEffect`

BugReports `https://github.com/marcozanotti/dispositionEffect/issues`

Depends R (>= 3.5.0)

NeedsCompilation no

Author Lorenzo Mazzucchelli [aut], Marco Zanotti [aut, cre]

Maintainer Marco Zanotti <`zanottimarco17@gmail.com`>

Repository CRAN

Date/Publication 2022-05-30 07:50:02 UTC

R topics documented:

`dispositionEffect-package` *dispositionEffect: behavioral Analysis on Financial Data*

Description

The dispositionEffect package allows to perform different types of behavioral analysis using financial market and experimental data. The analysis of disposition effect, portfolio-driven disposition effect, and time series disposition effect can be performed with four different implemented methods.

Main functions

- `portfolio_compute` is a wrapper function that compute realized and paper gains and losses from the investor's transactions and the market prices of the traded assets and updates the investor's portfolio
- `gains_losses` is the core function of the package. It performs all the necessary calculations and can be used for real-time processing (it is intended for advanced users only)
- `disposition_effect` Compute the disposition effect based on realized and paper gains and losses
- `disposition_difference` Compute the disposition difference based on realized gains and losses
- `disposition_compute`and `disposition_summary`interfaces that allow to easily compute disposition effect and summary statistics.

Author(s)

L. Mazzucchelli & M. Zanotti

References

- An, Li and Engelberg, Joseph and Henriksson, Matthew and Wang, Baolian and Williams, Jared, 2019, "The Portfolio-Driven Disposition Effect".
- Filippin, Mazzucchelli and Zanotti, 2021, "An analysis of the short selling impact on the disposition effect extended in the portfolio framework" (working paper).
- Mazzucchelli, 2021, "An Analysis of Short Selling and Volatility Impact on the Disposition Effect" (working paper).
- Odean, Terrance, 1998, "Are investors reluctant to realize their losses?" Journal of Finance 53:5, 1775-98.
- Sakaguchi, Hiroaki and Stewart, Neil and Walasek, Lukasz, 2019, "The Disposition Effect Varies with Portfolio Composition Because People Take Gain-Loss-Domain-Level Sell Decisions".
- Shefrin, Hersh, and Meir Statman, 1985, "The disposition to sell winners too early and ride losers too long", Journal of Finance 40:3, 777-90.
- Weber, Martin, and Colin F. Camerer, 1998, "The disposition effect in securities trading: An experimental analysis", Journal of Economic Behavior and Organization 33:2, 167-84.

`closest_market_price` *Closest market price*

Description

Find the market price closest to a certain datetime and for specific assets.

Usage

```
closest_market_price(
  asset,
  datetime,
  market_prices,
  price_only = FALSE,
  exact = FALSE,
  substitute_datetime = FALSE
)
```

Arguments

`asset`	Character vector of assets' names to look for.
`datetime`	POSIXct of the datetime at which looking for the asset's price.
`market_prices`	Data frame containing the market prices.
`price_only`	Logical. If TRUE then only the price is returned.
`exact`	Logical. If TRUE then it looks for market prices at the same datetime only, otherwise it looks for the nearest before the datetime.
`substitute_datetime`	Logical. If TRUE the datetime is substituted with the datetime argument.

Value

The data frame of closest market prices.

See Also

`evaluate`, `lubridate::round_date`

`DEanalysis` *Real sample data for Disposition Effect analysis*

Description

A sample dataset containing 10 investors, their market transactions and the market prices of the traded assets.

Usage

```
DEanalysis
```

Format

A list containing two data frames: transactions and marketprices.

investor id of the investor

type binary variable indicating the type of operation, B = buy and S = sell

asset id of the traded asset

quantity quantity of the traded asset

price market price of the traded asset

datetime timestamp of the operation

`disposition_effect` *Disposition Effect*

Description

Compute the disposition effect and the disposition difference.

Usage

```
disposition_effect(realized_gains, paper_gains, realized_losses, paper_losses)

disposition_difference(gains, losses)

disposition_compute(
  gainslosses,
  dispdiff_value = FALSE,
  aggregate_fun = NULL,
  ...
)

disposition_compute_ts(gainslosses, aggregate_fun = NULL, ...)

disposition_summary(gainslosses, dispdiff_value = FALSE)

disposition_summary_ts(de_timeseries)
```

Arguments

`realized_gains`	Numeric vector (or scalar) containing realized gains values.
`paper_gains`	Numeric vector (or scalar) containing paper gains values.
`realized_losses`	Numeric vector (or scalar) containing realized losses values.
`paper_losses`	Numeric vector (or scalar) containing paper losses values.
`gains`	Numeric vector (or scalar) containing gains.
`losses`	Numeric vector (or scalar) containing losses.
`gainslosses`	Data frame, the portfolio of the investor containing the realized and paper gains and losses results (as those obtained via `portfolio_compute`).
`dispdiff_value`	Logical, if TRUE the disposition difference on the "value" method is computed. Default to disposition effect (FALSE).
`aggregate_fun`	Function to use to aggregate results. Default to `NULL`, that is no aggregation is performed and the results of each asset are shown.
`...`	Further arguments to be passed to the aggregate function.
`de_timeseries`	Data frame, the time series of disposition effects.

Details

The disposition effect is defined as $DE = (RealizedGain/(RealizedGain - PaperGain)) - (RealizedLoss/(RealizedLoss + PaperLoss))$

The disposition difference is defined as $DD = RealizedGain - |RealizedLoss|$ or $DD = PaperGain - |PaperLoss|$

Value

Numeric vector (or scalar) with the value(s) of disposition effect(s) or disposition difference(s).

Functions

- `disposition_effect`: Compute the disposition effect
- `disposition_difference`: Compute the disposition difference
- `disposition_compute`: Compute the disposition effect directly on the investor's portfolio containing realized and paper gains and losses results.
- `disposition_compute_ts`: Compute the time series disposition effect on the gains and losses results.
- `disposition_summary`: Wrapper that returns the most important summary statistics related to the disposition effect.
- `disposition_summary_ts`: Wrapper that returns the most important summary statistics related to the time series disposition effect.

`evaluate` *Portfolio evaluation*

Description

Calculate the portfolio value as the sum of each asset portfolio quantity times the excess return of each asset with respect to the market.

Usage

```
evaluate_portfolio(portfolio, market_prices)
```

Arguments

`portfolio`	Data frame of the investor's portfolio at time t.
`market_prices`	Data frame containing the market prices.

Value

The portfolio value.

See Also

`portfolio_compute`, `gains_losses`, `closest_market_price`

gains_losses *Gains & Losses*

Description

Calculation of the realized gains and losses and the paper gains and losses.

Usage

```
gains_losses(
  portfolio,
  market_prices,
  transaction_type,
  transaction_asset,
  transaction_quantity,
  transaction_price,
  transaction_datetime,
  previous_datetime,
  time_threshold = "0 mins",
  method = "all",
  allow_short = FALSE,
  verbose = FALSE
)
```

Arguments

`portfolio`	Data frame of the investor's portfolio at time t.
`market_prices`	Data frame containing the market prices.
`transaction_type`	Character string. Either "B" = buy or "S" = sell.
`transaction_asset`	Character string. The name of the traded asset.
`transaction_quantity`	Numeric value. The quantity of the traded asset.
`transaction_price`	Numeric value. The market price of the traded asset.
`transaction_datetime`	POSIXct value. The date-time at which the transaction is going to occur.
`previous_datetime`	POSIXct value. The date-time of the last transaction performed by the investor.
`time_threshold`	Character in the format "value units" indicating the time threshold at which the computed financial difftime has to be evaluated (for instance "05 mins" or "20 hours"). The allowed units are "secs", "mins", "hours", "days" and "weeks" (See `base::difftime`).

`method`	Character string. The method used to compute papers. Allowed values are "count", "total", "value", "duration" and "all".
`allow_short`	Logical. If TRUE short positions are allowed, otherwise only long positions are allowed.
`verbose`	Logical. If TRUE than messages are printed to the console.

Details

It is essentially a wrapper around `paper_compute` and `realized_compute` functions. It is the function that can be used for streaming computations of gains and losses.

Value

A data frame containing the values of realized and paper gains and losses computed by means of the chosen method on each portfolio assets.

See Also

`realized_compute`, `paper_compute`, `portfolio_compute`

`investor` *Sample investor financial transactions*

Description

A sample dataset containing 19 transactions over time.

Usage

```
investor
```

Format

A data frame with 19 rows and 6 variables:

investor id of the investor

type binary variable indicating the type of operation, B = buy and S = sell

asset id of the traded asset

quantity quantity of the traded asset

price market price of the traded asset

datetime timestamp of the operation

`marketprices` *Market prices of assets traded by the sample investor*

Description

A sample dataset containing 6895 market prices of 5 different assets over time.

Usage

```
marketprices
```

Format

A data frame with 6895 rows and 4 variables:

asset id of the asset

datetime timestamp of market price

price market price of the asset

`paper_compute` *Papers' estimation*

Description

Compute paper gains and paper losses as either simple counts, total quantities, expected returns and financial duration.

Usage

```
paper_count(
  portfolio_quantity,
  portfolio_price,
  market_price,
  allow_short = TRUE
)

paper_total(
  portfolio_quantity,
  portfolio_price,
  market_price,
  allow_short = TRUE
)

paper_value(
  portfolio_quantity,
  portfolio_price,
  market_price,
  allow_short = TRUE
)

paper_duration(
  portfolio_quantity,
  portfolio_price,
  market_price,
  datetime_difference = NULL,
  previous_datetime = NULL,
  transaction_datetime = NULL,
  allow_short = TRUE
)

paper_compute(
  portfolio_quantity,
  portfolio_price,
  market_price,
  previous_datetime,
  transaction_datetime,
  assets,
  allow_short = TRUE,
  method = "all"
)
```

Arguments

`portfolio_quantity`	Numeric vector. The portfolio quantities of assets into the investor's portfolio.
`portfolio_price`	Numeric vector. The portfolio prices of assets into the investor's portfolio.
`market_price`	Numeric vector. The market prices of assets into the investor's portfolio.
`allow_short`	Logical. If TRUE short positions are allowed, otherwise only long positions are allowed.
`datetime_difference`	Numeric value of time difference between the previous_datetime and the transaction_datetime, computed through `difftime_financial`. If NULL, then previous_datetime and transaction_datetime must be specified.
`previous_datetime`	POSIXct value. The date-time of the last transaction performed by the investor.
`transaction_datetime`	POSIXct value. The date-time at which the transaction is going to occur.
`assets`	Character vector. The name of assets into the investor's portfolio but the traded asset.
`method`	Character string. The method used to compute papers. Allowed values are "count", "total", "value", "duration" and "all".

Value

The described functions have different return behaviors

- `paper_compute` returns a data frame containing the values of paper gains and paper losses computed by means of the chosen method on each portfolio assets.
- `paper_count` returns a named vector containing the values of paper gains and paper losses computed using the count method.
- `paper_total` returns a named vector containing the values of paper gains and paper losses computed using the total method.
- `paper_value` returns a named vector containing the values of paper gains and paper losses computed using the value method.
- `paper_duration` returns a named vector containing the values of paper gains and paper losses computed using the duration method.

In particular:

- `RG_"method"` contains Realized Gains results
- `RL_"method"` contains Realized Losses results
- `PG_"method"` contains Paper Gains results
- `PL_"method"` contains Paper Losses results

Functions

- `paper_count`: Computation of paper gains and paper losses as simple counts (default method).
- `paper_total`: Computation of paper gains and paper losses as total quantity of assets.
- `paper_value`: Computation of paper gains and paper losses as expected return of assets.
- `paper_duration`: Computation of paper gains and paper losses as financial duration.
- `paper_compute`: Wrapper that calls other paper_. functions to compute paper gains and paper losses based on the chosen method.

See Also

`realized_compute`, `gains_losses`

`portfolio_compute` *Portfolio Compute*

Description

Computation of all the transaction updates and the realized and paper gains and losses for each assets.

Usage

```
portfolio_compute(
  portfolio_transactions,
  market_prices,
  method = "count",
  allow_short = TRUE,
  time_threshold = "0 mins",
  exact_market_prices = TRUE,
  portfolio_driven_DE = FALSE,
  time_series_DE = FALSE,
  assets_time_series_DE = NULL,
  verbose = c(0, 0),
  progress = FALSE
)
```

Arguments

`portfolio_transactions`	Data frame. The investor's transactions data frame.
`market_prices`	Data frame containing the market prices.
`method`	Character string containing the method to use to compute realized and paper gains and losses. If "none" nothing is computed but the investor's portfolio updates. Otherwise it has to be one of "count" (default), "total", "value", "duration", or "all".
`allow_short`	Logical. If TRUE short positions are allowed, otherwise only long positions are allowed.
`time_threshold`	Character in the format "value units" indicating the time threshold at which the computed financial difftime has to be evaluated (for instance "05 mins" or "20 hours"). The allowed units are "secs", "mins", "hours", "days" and "weeks" (See `base::difftime`).
`exact_market_prices`	Logical. If TRUE then `closest_market_price` uses exact datetime match to look for the closest price of each asset. It usually speeds up computation by a small degree, but it requires the `market_prices` to have the prices for each transaction asset along each transaction datetimes.
`portfolio_driven_DE`	Logical. If TRUE the realized and paper gains and losses for the positive (that is when the investor's portfolio value, as computed through `evaluate_portfolio`, is greater than zero) and the negative (that is when the investor's portfolio value, as computed through `evaluate_portfolio`, is smaller than zero) portfolios are returned.
`time_series_DE`	Logical. If TRUE the time series of disposition effect is computed on 'count' and 'value' methods only.
`assets_time_series_DE`	Character vector of assets' names as contained into `portfolio_transactions` on which to compute the time series disposition effect.

`verbose`	Numeric or logical vector of length 2 that allows to control for the function's verbosity.
`progress`	Logical. If TRUE a progress bar is displayed.

Value

A data frame containing the investor's portfolio and the values of realized and paper gains and losses computed by means of the chosen method on each portfolio assets.

If time_series_DE is set to TRUE, then also time series disposition effect results are returned.

See Also

`realized_compute`, `paper_compute`, `gains_losses`

`portfolio_results` *Realized and paper results*

Description

Results obtained by means of `portfolio_compute` on the data sets `investor` and `marketprices`.

Usage

```
portfolio_results
```

Format

A data frame with 5 rows and 21 variables:

investor id of the investor

asset id of the traded asset

quantity quantity of the traded asset at the end of the portfolio updating process

price last market price of the traded asset

datetime timestamp of the last operation

RG_count realized gains via count method

RL_count realized losses via count method

PG_count paper gains via count method

PL_count paper losses via count method

RG_total realized gains via total method

RL_total realized losses via total method

PG_total paper gains via total method

PL_total paper losses via total method

RG_value realized gains via value method

RL_value realized losses via value method

PG_value paper gains via value method

PL_value paper losses via value method

RG_duration realized gains via duration method

RL_duration realized losses via duration method

PG_duration paper gains via duration method

PL_duration paper losses via duration method

`portfolio_results_ts` *Realized and paper results*

Description

Results obtained by means of `portfolio_compute` on the data sets `investor` and `marketprices` with `time_series_DE = TRUE`.

Usage

```
portfolio_results_ts
```

Format

A data frame with 19 rows and 6 variables:

investor id of the investor

datetime timestamp of the last operation

DEts_count Partial disposition effect computed at time t

DETs_count Complete disposition effect computed after updating at time t

DEts_value Partial disposition difference computed at time t

DETs_value Complete disposition difference computed after updating at time t

realized_compute *Realized estimation*

Description

Compute realized gains and realized losses as either simple counts, total quantities, expected returns and financial duration.

Usage

```
realized_count(
  portfolio_quantity,
  portfolio_price,
  transaction_quantity,
  transaction_price,
  transaction_type,
  allow_short = TRUE,
  realized_only = FALSE
)

realized_total(
  portfolio_quantity,
  portfolio_price,
  transaction_quantity,
  transaction_price,
  transaction_type,
  allow_short = TRUE,
  realized_only = FALSE
)

realized_value(
  portfolio_quantity,
  portfolio_price,
  transaction_quantity,
  transaction_price,
  transaction_type,
  allow_short = TRUE,
  realized_only = FALSE
)

realized_duration(
  portfolio_quantity,
  portfolio_price,
  transaction_quantity,
  transaction_price,
  transaction_type,
  previous_transaction_datetime,
  previous_datetime,
  transaction_datetime,
  allow_short = TRUE,
  realized_only = FALSE
)

realized_compute(
  portfolio_quantity,
  portfolio_price,
  transaction_quantity,
  transaction_price,
  transaction_type,
  previous_transaction_datetime,
  previous_datetime,
  transaction_datetime,
  transaction_asset,
  allow_short = TRUE,
  realized_only = FALSE,
  method = "all"
)

realized_empty(transaction_asset, method = "all")
```

Arguments

`portfolio_quantity`	Numeric vector. The portfolio quantities of assets into the investor's portfolio.
`portfolio_price`	Numeric vector. The portfolio prices of assets into the investor's portfolio.
`transaction_quantity`	Numeric value. The quantity of the traded asset.
`transaction_price`	Numeric value. The market price of the traded asset.
`transaction_type`	Character string. Either "B" = buy or "S" = sell.
`allow_short`	Logical. If TRUE short positions are allowed, otherwise only long positions are allowed.
`realized_only`	Logical. If TRUE only realized gains and realized losses are computed. Otherwise also paper gains and paper losses on excess quantity of the traded asset are computed.
`previous_transaction_datetime`	POSIXct value. The portfolio date-time related to the last transaction of the traded asset.
`previous_datetime`	POSIXct value. The date-time of the last transaction performed by the investor.
`transaction_datetime`	POSIXct value. The date-time at which the transaction is going to occur.

`transaction_asset`	Character string. The name of the traded asset.
`method`	Character string. The method used to compute papers. Allowed values are "count", "total", "value", "duration" and "all".

Value

The described functions have different return behaviors

- `realized_compute` returns a data frame containing the values of realized and paper gains and losses computed by means of the chosen method on each portfolio assets.
- `realized_count` returns a named vector containing the values of realized and paper gains and losses computed using the count method.
- `realized_total` returns a named vector containing the values of realized and paper gains and losses computed using the total method.
- `realized_value` returns a named vector containing the values of realized and paper gains and losses computed using the value method.
- `realized_duration` returns a named vector containing the values of realized and paper gains and losses computed using the duration method.
- `realized_empty` returns a named vector containing empty values of realized and paper gains and losses computed using the chosen method.

In particular:

- `RG_"method"` contains Realized Gains results
- `RL_"method"` contains Realized Losses results
- `PG_"method"` contains Paper Gains results
- `PL_"method"` contains Paper Losses results

Functions

- `realized_count`: Computation, as simple counts, of realized gains and realized losses of the traded asset.
- `realized_total`: Computation, as total quantity, of realized gains and realized losses of the traded asset.
- `realized_value`: Computation, as expected return, of realized gains and realized losses of the traded asset.
- `realized_duration`: Computation, as financial duration, of realized gains and realized losses of the traded asset.
- `realized_compute`: Wrapper that calls other realized_. functions to compute realized gains and realized losses of the traded asset based on the chosen method.
- `realized_empty`: Simple function to obtain empty results for realized and paper computations based on the chosen method.

See Also

`paper_compute`, `gains_losses`

Index