Package 'EventStudy'

March 14, 2019

Type Package

Title Event Study Analysis

Description Perform Event Studies from through our <http://EventStudyTools.com> Application Programming Interface, parse the results, visualize it, and / or use the results in further analysis.

Author Dr. Simon Mueller

Date 2019-03-05

Version 0.36

Encoding UTF-8

URL http://eventstudytools.com

Maintainer Dr. Simon Mueller <simon.mueller@muon-stat.com>

License GPL (>= 2)

Depends ggplot2

Imports shiny, miniUI, rstudioapi, httr, curl, jsonlite, magrittr (>= 1.5), data.table, testthat, dplyr, tidyr, rlang, scales, tidyquant, RColorBrewer, stringr, purrr, readr, openxlsx

Suggests knitr, rmarkdown

BugReports https://github.com/EventStudyTools/api-wrapper.r/issues

LazyData true

RoxygenNote 6.1.1

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2019-03-14 12:13:31 UTC

R topics documented:

`aarPlot` *Averaged Abnormal Return Plot*

Description

Averaged abnormal return plots with confidence intervals

For more details see the help vignette: `vignette("parameters_eventstudy", package = "EventStudy")`

Usage

```
aarPlot(ResultParserObj, cumSum = F, group = NULL, window = NULL,
  ciStatistics = NULL, p = 0.95, ciType = "band", xlab = "",
  ylab = "Averaged Abnormal Returns", facet = T, ncol = 4)
```

Arguments

`ResultParserObj`	An object of class `ResultParser`
`cumSum`	plot CAAR
`group`	set this parameter if just one group should be plotted
`window`	numeric vector of length 2
`ciStatistics`	Statistic used for confidence intervals
`p`	p-value
`ciType`	type of CI band
`xlab`	x-axis label
`ylab`	y-axis label
`facet`	should each firm get its own plot (default = T)
`ncol`	number of facet columns

Value

a ggplot2 object

Examples

```
## Not run:
# plot averaged abnormal returns in one plot
aarPlot(resultParser)

# plot averaged abnormal returns with .95-CI
arPlot(resultParser, ciStatistics = "Patell Z",p = .95)

## End(Not run)
```

`ARCApplicationInput` *Abnormal Return Calculation Parameters*

Description

This R6 class defines the parameters for the Return Event Study. We recommend to use the `set` functionality to setup your Event Study, as we check input parameters.

For more details see the help vignette: `vignette("parameters_eventstudy", package = "EventStudy")`

Usage

```
ARCApplicationInput
```

Format

An object of class `R6ClassGenerator` of length 24.

Value

a ESTParameters R6 object

Methods

`$new()` Constructor for ARCApplicationInput.

`$setEMail(eMail)` Set the e-Mail address for reporting. This functionality is currently not working.

`$setBenchmarkModel(model = 'mm')` Setter for the benchmark model.s

`$setReturnType(returnType)` Setter for the return type (log or simple)

`$setTestStatistics(testStatistics)` Setter for the test statistics.

Arguments

ESTARCParameters An `ARCApplicationInput` object

eMail An E-Mail address in `String` format

model A benchmark model in `String` format

returnType A return type in `String` format

testStatistics A `String` vector with test statistics.

See Also

https://www.eventstudytools.com/axc/upload

Examples

```
## Not run:
# get files for our S&P500 example; 3 files are written in the current
# working directory
getSP500ExampleFiles()

# Generate a new parameter object
arcParams <- ARCApplicationInput$new()

# set test statistics
arcParams$setBenchmarkModel("garch")

# Setup API object
apiKey <- "{Your API key}"
estSetup <- EventStudyAPI$new()
estSetup$authentication(apiKey)

# Perform Event Study
estSetup$performEventStudy(estParams = arcParams,
                           dataFiles = c("request_file" = "01_RequestFile.csv",
                                         "firm_data"    = "02_firmData.csv",
                                         "market_data"  = "03_marketData.csv"))

# Download task results and save them in the actiual working directory
estSetup$getTaskResults()

## End(Not run)
```

arPlot *Abnormal Return Plot*

Description

Plot abnormal returns in the event window of single or multiple firms.

Usage

```
arPlot(ResultParserObj, firm = NULL, window = NULL, xlab = "",
  ylab = "Abnormal Returns", alpha = 0.5, facetVar = NULL,
  ncol = 4, addAAR = F, xVar = "eventTime", yVar = "ar")
```

Arguments

`ResultParserObj`	An object of class `ResultParser`
`firm`	set this parameter if just a subset of firms should be plotted
`window`	filter event time window
`xlab`	x-axis label of the plot
`ylab`	y-axis label
`alpha`	alpha value
`facetVar`	should each firm get its own plot. You may plot each firm in an own plot or by each group. (Default: NULL, available: Group and Firm)
`ncol`	number of facet columns
`addAAR`	add aar line
`xVar`	x variable name
`yVar`	y variable name

Value

a ggplot2 object

Examples

```
## Not run:
# plot abnormal returns in one plot
arPlot(resultParser)

# plot abnormal returns by group
arPlot(resultParser, facetVar = "Group")

## End(Not run)
```

`AVCApplicationInput` *Abnormal Volume Calculation Parameters*

Description

This R6 class defines the parameters for the Abnormal Volume Event Study. We recommend to use the `set` functionality to setup your Event Study, as we check input parameters.

For more details see the help vignette: `vignette("parameters_eventstudy", package = "EventStudy")`

Usage

```
AVCApplicationInput
```

Format

R6Class object.

Value

a ESTParameters R6 object

Methods

`$new()` Constructor for AVCApplicationInput

`$setEMail(eMail)` Set the e-Mail address for reporting. This functionality is currently not working

`$setBenchmarkModel(model = ’mm’)` Setter for the benchmark models

`$setReturnType(returnType)` Setter for the return type (log or simple)

`$setTestStatistics(testStatistics)` Setter for the test statistics

Arguments

AVCApplicationInput An `AVCApplicationInput` object

eMail An E-Mail address in `String` format

model A benchmark model in `String` format

returnType A return type in `String` format

testStatistics A `String` vector with test statistics

See Also

https://www.eventstudytools.com/axc/upload

Examples

```
## Not run:
# get files for our S&P500 example; 3 files are written in the current
# working directory
getSP500ExampleFiles()

# Generate a new parameter object
avcParams <- AVCApplicationInput$new()

# set test statistics
arcParams$setBenchmarkModel(”garch”)

# Setup API object
apiKey <- "{Your API key}”
estSetup <- EventStudyAPI$new()
estSetup$authentication(apiKey)

# Perform Event Study
estSetup$performEventStudy(estParams = avcParams,
                                dataFiles = c("request_file" = "01_RequestFile.csv",
                                              "firm_data"    = "02_firmData.csv",
                                              "market_data"  = "03_marketData.csv"))

    # Download task results and save them in the actiual working directory
    estSetup$getTaskResults()

    ## End(Not run)
```

AVyCApplicationInput *Abnormal Volatility Calculation Parameters*

Description

This R6 class defines the parameters for the Abnormal Volatility Volume Event Study. We recommend to use the `set` functionality to setup your Event Study, as we check input parameters.

For more details see the help vignette: `vignette("parameters_eventstudy", package = "EventStudy")`

Usage

```
AVyCApplicationInput
```

Format

`R6Class` object.

Value

a ESTParameters R6 object

Methods

`$new()` Constructor for AVyCApplicationInput

`$setEMail(eMail)` Set the e-Mail address for reporting. This functionality is currently not working.

`$setBenchmarkModel(model = 'mm')` Setter for the benchmark models

`$setReturnType(returnType)` Setter for the return type (log or simple)

`$setTestStatistics(testStatistics)` Setter for the test statistics. Per default all available test statistics are applied. You may find all test statistics in the vignette 'parameter_eventstudy'

Arguments

AVyCApplicationInput An `AVyCApplicationInput` object

eMail An E-Mail address in `String` format

model A benchmark model in `String` format

returnType A return type in `String` format

testStatistics A `String` vector with test statistics

See Also

https://www.eventstudytools.com/axc/upload

Examples

```
## Not run:
# get files for our S&P500 example; 3 files are written in the current
# working directory
getSP500ExampleFiles()

# Generate a new parameter object
avycParams <- AVyCApplicationInput$new()

# set test statistics
avycParams$setTestStatistics(c("aarptlz", "aarrankz"))

# Setup API object
apiKey <- "{Your API key}"
estSetup <- EventStudyAPI$new()
estSetup$authentication(apiKey)

# Perform Event Study
estSetup$performEventStudy(estParams = avycParams,
                           dataFiles = c("request_file" = "01_RequestFile.csv",
                                         "firm_data"    = "02_firmData.csv",
                                         "market_data"  = "03_marketData.csv"))

# Download task results and save them in the actiual working directory
estSetup$getTaskResults()

## End(Not run)
```

checkFile *Check input data files*

Description

Check correct column, date, and shape of the input data files

Usage

```
checkFile(path, type = "request_file")
```

Arguments

path	path to the input data file
type	the type of file to ckeck

Value

data.frame

Examples

```
## Not run:
# save example files to current working directory
getSP500ExampleFiles()

checkFile("01_RequestFile.csv", "request_file")

## End(Not run)
```

checkFiles   *Check EventStudy input files*

Description

Check each input file plus inter file relations, whether market index and firm identifier in request file match market index in market_data and firm identifier in in firm_data file.

Usage

```
checkFiles(dataFiles = c(request_file = "01_RequestFile.csv", firm_data =
  "02_firmData.csv", market_data = "03_MarketData.csv"), returnData = F)
```

Arguments

dataFiles	A named character vector. The names must be request_file, firm_data, and market_data
returnData	returns the data as list of data.frames

Examples

```
## Not run:
# save example files to current working directory
getSP500ExampleFiles()

dataFiles <- c("request_file" = "01_RequestFile.csv",
               "firm_data"    = "02_firmData.csv",
               "market_data"  = "03_MarketData.csv")

checkFiles(dataFiles)

## End(Not run)
```

`estAPIKey` *Set eventStudy API Key*

Description

Set eventStudy API Key

Usage

```
estAPIKey(key)
```

Arguments

`key`	EventStudy API Key

`EventStudy` *EventStudy*

Description

This package provides functionality for doing Event Studies from R by using EventStudyTools.com API interface, parsing results, and visualize them.

Details

To learn more about EventStudy visit project website: `www.eventstudytools.com` or start with the vignettes: `browseVignettes(package = "EventStudy")`

`EventStudyAddin` *RStudio Addin for performing an Event Study*

Description

Call this as an addin to perform an Event Study on an interface in R. The interface is similar to our Event Study web interface `www.eventstudytools.com`.

Usage

```
EventStudyAddin()
```

EventStudyAPI *API for* www.eventstudytools.com

Description

R interface for performing Event Studies on www.eventstudytools.com. Please get a free API key from our website: https://www.eventstudytools.com/api-key.

For more details see the help vignette: `vignette("introduction_eventstudy", package = "EventStudy")`

Usage

```
EventStudyAPI
```

Format

R6Class object

Usage

For usage details see **Methods, Arguments, and Examples** sections.

Methods

`new(apiServerUrl)` This method is used to create an object of this class with `apiServerUrl` as the url to the EventStudyTools server

`authentication(apiKey)` This method is used to authenticate at `apiServerUrl`. A valid `APIkey` is required. You can download a free key on our website: www.eventstudytools.com

`performEventStudy(estParam)` This method starts an Event Study. This method does all the analysis work for you

`performDefaultEventStudy()` This method starts a default Event Study. It is a wrapper around `performEventStudy`

`processTask()` This method starts the Event Study calculation on the server (after files are uploaded.

`configureTask(input)` This method configures the Event Study. `input` is an `ApplicationInputInterface` R6 object, e.g. ARC configuration class

`uploadFile(fileKey, fileName)` This method links to the file to upload. `fileKey` is the key of the file. Valid values are: `request_file`, `firm_data`, and `market_data`. `fileName` file name to upload.

`commitData()` This method commits the data to the server

`getTaskStatus()` Check if calculation is finished

`getTaskResults(destDir = getwd())` Downloads the result files of the Event Study to `destDir` (Default: current working directory).

Arguments

eventstudyapi An `EventStudyAPI` object

apiServerUrl URL to the API endpoint

apiKey Key for authentication

input An `ApplicationInputInterface` object.

fileKey Type of input file: `request_file`, `firm_data`, and `market_data`

fileName Data filename

destDir Directory for saving result files

Examples

```
## Not run:
apiKey <- "{Please insert your aPI key here}"

The URL is already set by default
options(EventStudy.KEY = apiKey)

# initialize object
estSetup <- EventStudyAPI$new()

# get S&P500 example data
getSP500ExampleFiles()

# set Event Study parameters
estType <- "arc"
dataFiles <- c("request_file" = "01_RequestFile.csv",
               "firm_data"    = "02_firmData.csv",
               "market_data"  = "03_MarketData.csv")
resultPath <- "results"

# Perform Event Study
estResult <- estSetup$performDefaultEventStudy(estType    = estType,
                                               dataFiles  = dataFiles,
                                               destDir    = resultPath)

## End(Not run)
```

`EventStudyApplicationInput`

Abnormal Return Calculation (ARC) API Wrapper

Description

This R6 class serialzes an Event Study parameter class to a list structure. This is an abstract class for Event Study applications (Return, Volatility, and Volume Event Studies). It is not intended to use this class directly. Please use: ARCApplicationInput.

Usage

```
EventStudyApplicationInput
```

Format

`R6Class` object.

Methods

`$new()` Constructor for EventStudyApplicationInput

`$setup()` Setup the parameter list

`getSP500ExampleFiles` *This function copies the three csv files to the actual working directory. This example data is used as motivation for using Event Studies for Additions / Deletions to market indices.*

Description

For more details see the help vignette: `vignette("introduction_eventstudy", package = "EventStudy")`

Usage

```
getSP500ExampleFiles(targetDir = getwd())
```

Arguments

`targetDir`	directory to save example files

Details

or on our website: https://www.eventstudytools.com/mergers-acquisitions

Examples

```
## Not run:
getSP500ExampleFiles("data")

## End(Not run)
```

ResultParser *Parses request and results files returned from our Event Study API interface.*

Description

This result file parser works currently only with csv files. Please read the vignette for further details (coming soon). We will restructure our result reports soon. So, this function may change dramatically. This object can be used for plotting your results.

Usage

```
ResultParser
```

Format

R6Class object.

Methods

`new(dir)` This method is used to create object of this class with `dir` as the directory of result files.

`parseReport(path = "analysis_report.csv")` This method parses the analysis report file (analysis_report.csv).

`parseAR(path = "ar_results.csv")` This method parses the abnormal return file (ar_results.csv). Furthermore, it triggers `parseReport` and join firm and index name.

`parseCAR(path = "car_results.csv")` This method parses the cumulative abnormal return file (ar_results.csv). Furthermore, it triggers `parseReport` and join firm and index name.

Examples

```
## Not run:
# Assume you already performed an Event Study and result files are saved in
# the actual working directory.
estParser <- ResultParser$new()

# parse request file
estParser$parseRequestFile("01_RequestFile.csv")

# parse result files
estParser$parseReport("Analysis report.csv")
estParser$parseAR("AR results.csv")
estParser$parseAAR("AAR results.csv")

## End(Not run)
```

Index