

Package ‘EventStudy’

October 12, 2022

Type Package

Title Event Study Analysis

Description

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

Author Dr. Simon Mueller

Date 2022-09-17

Version 0.39

Encoding UTF-8

URL <https://data-zoo.de>

Maintainer Dr. Simon Mueller <sm@data-zoo.de>

License MIT + file LICENSE

Depends ggplot2

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

Suggests knitr, rmarkdown

BugReports <https://github.com/sipemu/eventstudy/issues>

RoxygenNote 7.2.1

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2022-09-20 12:45:50 UTC

R topics documented:

aar_results	2
ARCAApplicationInput	6
AVCAApplicationInput	9

AVyCApplicationInput	11
CAAR Results.	12
checkFile	13
checkFiles	14
estAPIKey	15
EventStudy	15
EventStudyAddin	16
EventStudyAPI	16
EventStudyApplicationInput	21
getSP500ExampleFiles	22
ResultParser	22

Index	25
--------------	-----------

aar_results	<i>An R6 object that contains AAR results.</i>
-------------	--

Description

An R6 object that contains AAR results.

An R6 object that contains AAR results.

An R6 object that contains AAR results.

An R6 object that contains AAR results.

An R6 object that contains AAR results.

An R6 object that contains AAR results.

Format

[R6Class](#) object.

[R6Class](#) object.

[R6Class](#) object.

Methods

plot This method plots aar results.

plot This method plots aar results.

plot This method plots aar results.

Public fields

aar_tbl AAR results.

statistics_tbl AAR test statistic results.

Methods

Public methods:

- `AARResults$new()`
- `AARResults$print()`
- `AARResults$plot()`
- `AARResults$plot_cumulative()`
- `AARResults$confidence_interval()`
- `AARResults$plot_test_statistics()`
- `AARResults$clone()`

Method `new()`: Class initialization

Usage:

```
AARResults$new(aar_tbl, statistics_tbl)
```

Arguments:

`aar_tbl` AAR result table.

`statistics_tbl` Table with statistics.

Method `print()`: Print key characteristics.

Usage:

```
AARResults$print()
```

Method `plot()`: Plots AAR results for each analysis group.

Usage:

```
AARResults$plot(
  group = NULL,
  ci_statistics = NULL,
  p = 0.95,
  ci_type = "two-sided",
  xlab = "Event Window",
  ylab = "Averaged Abnormal Returns",
  facet = T,
  ncol = 4
)
```

Arguments:

`group` Subset to your analysed groups, else all groups will be plotted.

`ci_statistics` Statistic used for confidence intervals

`p` The desired p-value

`ci_type` type of CI band for ggplot2, available are band or ribbon.

`xlab` x-axis label

`ylab` y-axis label

`facet` should each group get its own plot (default = T)

`ncol` number of facet columns

Method `plot_cumulative()`: Plot Cumulative Abnormal Return. No test statistic is available.

Usage:

```
AARResults$plot_cumulative(
  xlab = "Event Window",
  ylab = "Cumulative Averaged Abnormal Returns",
  facet = T,
  ncol = 4
)
```

Arguments:

xlab x axis lab
 ylab y axis lab
 facet Shall the plot faceted by Group
 ncol Number of cols when faceting.

Method `confidence_interval()`: Calculates Confidence band for given test statistic.

Usage:

```
AARResults$confidence_interval(
  statistic = "Patell Z",
  p = 0.95,
  ci_type = "two-sided"
)
```

Arguments:

statistic Chosen test statistics for calculation.
 p Chosen p value.
 ci_type Type of confidence interval.

Method `plot_test_statistics()`: Plots a heatmap with test statistics on y axis and Day Relative to Event on x axis. Colorization is done according to significance according to given p.

Usage:

```
AARResults$plot_test_statistics(p = 0.95, ci_type = "two-sided")
```

Arguments:

p Chosen p value.
 ci_type CI type.

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
AARResults$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Public fields

ar_tbl AR result table. Class initialization

Methods**Public methods:**

- [ARResults\\$new\(\)](#)
- [ARResults\\$print\(\)](#)
- [ARResults\\$plot\(\)](#)
- [ARResults\\$clone\(\)](#)

Method new():

Usage:

```
ARResults$new(ar_tbl)
```

Arguments:

ar_tbl AR result table.

Method print(): Print key characteristics.

Usage:

```
ARResults$print()
```

Method plot(): Plot abnormal returns in the event window of single or multiple firms.

Usage:

```
ARResults$plot(firm = NULL, xlab = "", ylab = "Abnormal Returns", addAAR = F)
```

Arguments:

firm set this parameter if just a subset of firms should be plotted

xlab x-axis label of the plot

ylab y-axis label

addAAR add aar line

Returns: a ggplot2 object

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
ARResults$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Public fields

car_tbl Car result table Class initialization

Methods**Public methods:**

- [CARResults\\$new\(\)](#)
- [CARResults\\$print\(\)](#)
- [CARResults\\$clone\(\)](#)

Method new():*Usage:*

CAResults\$new(car_tbl)

Arguments:

car_tbl CAR result table.

Method print(): Print key characteristics.*Usage:*

CAResults\$print()

Method clone(): The objects of this class are cloneable with this method.*Usage:*

CAResults\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

 ARCAApplicationInput *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")

Value

a ESTParameters R6 object

Methods

`$new()` Constructor for ARCAApplicationInput.

`$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

ESTARCPParameters An ARCAApplicationInput 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.

Super classes

```
EventStudy::ApplicationInputInterface -> EventStudy::EventStudyApplicationInput -  
> ARCAApplicationInput
```

Public fields

task Task description
key Key
benchmark_model Benchmark model
return_type Return type
non_trading_days How to handle non-trading days
test_statistics Test statistics
request_file Request file
firm_data Firm data
market_data Market data

Methods**Public methods:**

- [ARCAApplicationInput\\$setEMail\(\)](#)
- [ARCAApplicationInput\\$setBenchmarkModel\(\)](#)
- [ARCAApplicationInput\\$setReturntype\(\)](#)
- [ARCAApplicationInput\\$setNonTradingDays\(\)](#)
- [ARCAApplicationInput\\$setTestStatistics\(\)](#)
- [ARCAApplicationInput\\$setDataFiles\(\)](#)
- [ARCAApplicationInput\\$clone\(\)](#)

Method setEMail(): set email

Usage:

```
ARCAApplicationInput$setEMail(eMail)
```

Arguments:

eMail Your E-mail address

Method setBenchmarkModel(): set benchmark model

Usage:

```
ARCAApplicationInput$setBenchmarkModel(model)
```

Arguments:

model benchmark model

Method setReturntype(): Set return type

Usage:

```
ARCAApplicationInput$setReturntype(returnType)
```

Arguments:

returnType return type

Method setNonTradingDays(): Set non trading days*Usage:*

ARCAApplicationInput\$setNonTradingDays(nonTradingDays = "later")

Arguments:

nonTradingDays how to handle non traing days

Method setTestStatistics(): Set test statistics*Usage:*

ARCAApplicationInput\$setTestStatistics(testStatistics = NULL)

Arguments:

testStatistics Test statistic

Method setDataFiles(): Set reques, firm, and market data file*Usage:*

```
ARCAApplicationInput$setDataFiles(
  dataFiles = c(request_file = "01_RequestFile.csv", firm_data = "02_firmData.csv",
    market_data = "03_MarketData.csv")
)
```

Arguments:

dataFiles Named vector of data files.

Method clone(): The objects of this class are cloneable with this method.*Usage:*

ARCAApplicationInput\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

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 <- ARCAApplicationInput$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 actual working directory
estSetup$getTaskResults()

## End(Not run)

```

AVCAApplicationInput *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")`

Format

[R6Class](#) object.

Value

a ESTParameters R6 object

Methods

`$new()` Constructor for AVCAApplicationInput
`$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

AVCAApplicationInput An AVCAApplicationInput 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

Super classes

```
EventStudy::ApplicationInputInterface -> EventStudy::EventStudyApplicationInput -
> EventStudy::ARCAApplicationInput -> AVCApplicationInput
```

Public fields

key Key of the Parameter set.

Methods**Public methods:**

- `AVCApplicationInput$clone()`

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
AVCApplicationInput$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

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 actual 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")`

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

Super classes

[EventStudy::ApplicationInputInterface](#) -> [EventStudy::EventStudyApplicationInput](#) -> AVyCApplicationInput

Public fields

`key` Key of the Parameter set.

`test_statistics` Available test statistics.

Methods**Public methods:**

- [AVyCApplicationInput\\$clone\(\)](#)

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
AVyCApplicationInput$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

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 actual working directory
estSetup$getTaskResults()

## End(Not run)
```

CAAR Results.

An R6 object that contains CAAR results.

Description

An R6 object that contains CAAR results.

An R6 object that contains CAAR results.

Format

R6Class object.

Public fields

caar_tbl CAAR results.

statistics_tbl CAAR test statistic results. Class initialization

Methods**Public methods:**

- CAARResults\$new()
- CAARResults\$print()
- CAARResults\$clone()

Method new():

Usage:

```
CAARResults$new(caar_tbl, statistics_tbl)
```

Arguments:

caar_tbl CAAR result table.

statistics_tbl Table with statistics.

Method print(): Print key characteristics.

Usage:

```
CAARResults$print()
```

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
CAARResults$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

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	<i>Check EventStudy input files</i>
------------	-------------------------------------

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

Start with the vignettes: `browseVignettes(package = "EventStudy")`

EventStudyAddin	<i>RStudio Addin for performing an Event Study</i>
-----------------	--

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 <https://www.eventstudytools.com>.

Usage

```
EventStudyAddin()
```

EventStudyAPI	<i>APE Entry Point</i>
---------------	------------------------

Description

R interface for performing Event Studies on <https://www.eventstudytools.com>.

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

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: <https://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

Public fields

`resultFiles` list of result files

`dataFiles` list of data files

Methods

Public methods:

- `EventStudyAPI$new()`
- `EventStudyAPI$authentication()`
- `EventStudyAPI$performEventStudy()`
- `EventStudyAPI$performDefaultEventStudy()`
- `EventStudyAPI$processTask()`
- `EventStudyAPI$configureTask()`
- `EventStudyAPI$uploadFile()`
- `EventStudyAPI$deleteFileParts()`
- `EventStudyAPI$splitFile()`
- `EventStudyAPI$get_token()`
- `EventStudyAPI$commitData()`
- `EventStudyAPI$getTaskStatus()`
- `EventStudyAPI$getTaskResults()`
- `EventStudyAPI$getApiVersion()`
- `EventStudyAPI$clone()`

Method `new()`: Class initialization

Usage:

```
EventStudyAPI$new(apiServerUrl = NULL)
```

Arguments:

apiServerUrl url to API server

Method authentication():*Usage:*

```
EventStudyAPI$authentication(apiKey = NULL)
```

Arguments:

apiKey EST API key

Method performEventStudy(): Performs an event study with given parameters and files.*Usage:*

```
EventStudyAPI$performEventStudy(
  estParams = NULL,
  dataFiles = c(request_file = "01_RequestFile.csv", firm_data = "02_firmData.csv",
    market_data = "03_MarketData.csv"),
  destDir = "results",
  downloadFiles = T,
  checkFiles = F
)
```

Arguments:

estParams A class of type ARCAApplicationInput. This class contains the definition of the event study.

dataFiles A named vector for the input files.

destDir Destination dir of event study results.

downloadFiles Boolean parameter for downloading files from server.

checkFiles Check input files.

Method performDefaultEventStudy(): Performs an event study with default parameters and files.*Usage:*

```
EventStudyAPI$performDefaultEventStudy(
  estType = "arc",
  dataFiles = c(request_file = "01_RequestFile.csv", firm_data = "02_firmData.csv",
    market_data = "03_MarketData.csv"),
  destDir = "results",
  downloadFiles = T,
  checkFiles = F
)
```

Arguments:

estType A string (arc, avc, or avyc) that is used to initialize the default parameter set.

dataFiles A named vector for the input files.

destDir Destination dir of event study results.

downloadFiles Boolean parameter for downloading files from server.
checkFiles Check input files.

Method processTask(): Process the task. Internal use.

Usage:

EventStudyAPI\$processTask()

Method configureTask(): Configure the task. Internal usage.

Usage:

EventStudyAPI\$configureTask(estParams = NULL)

Arguments:

estParams An object of class EventStudyApplicationInput

Method uploadFile(): Upload files to server. Internal usage.

Usage:

EventStudyAPI\$uploadFile(fileKey, fileName, partNumber = 0)

Arguments:

fileKey File key

fileName File name

partNumber Part number of the file

Method deleteFileParts(): Delete files. Internal usage.

Usage:

EventStudyAPI\$deleteFileParts(parts)

Arguments:

parts Parts

Method splitFile(): Split files Internal usage.

Usage:

EventStudyAPI\$splitFile(fileName, maxChunkSize)

Arguments:

fileName File name

maxChunkSize Max chunk size.

Method get_token(): Get token. Internal usage.

Usage:

EventStudyAPI\$get_token()

Method commitData(): Commit data. Internal usage.

Usage:

EventStudyAPI\$commitData()

Method getTaskStatus(): Fetch task status. Internal usage.

Usage:

```
EventStudyAPI$getTaskStatus(exceptionOnError = FALSE)
```

Arguments:

exceptionOnError Throw exception on error.

Method `getTaskResults()`: Fetch results Internal usage.

Usage:

```
EventStudyAPI$getTaskResults(downloadFiles = T, destDir = getwd())
```

Arguments:

downloadFiles Download files

destDir Destination dir

Method `getApiVersion()`: Get API version.

Usage:

```
EventStudyAPI$getApiVersion()
```

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
EventStudyAPI$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

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 serializes 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: [ARCAApplicationInput](#).

Format

[R6Class](#) object.

Methods

`$new()` Constructor for EventStudyApplicationInput

`$setup()` Setup the parameter list

Super class

[EventStudy::ApplicationInputInterface](#) -> EventStudyApplicationInput

Methods

Public methods:

- [EventStudyApplicationInput\\$setup\(\)](#)
- [EventStudyApplicationInput\\$clone\(\)](#)

Method `setup()`: Initialize parameters of an event study

Usage:

`EventStudyApplicationInput$setup()`

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

`EventStudyApplicationInput$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

`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.

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.

Public fields

`destDir` Result dir.

Methods**Public methods:**

- `ResultParser$get_request_file()`
- `ResultParser$get_analysis_report()`
- `ResultParser$get_ar()`
- `ResultParser$get_car()`
- `ResultParser$get_aar()`
- `ResultParser$get_caar()`
- `ResultParser$clone()`

Method `get_request_file()`: Parse request file

Usage:

```
ResultParser$get_request_file(path = "01_RequestFile.csv")
```

Arguments:

`path` path to request file.

Method `get_analysis_report()`: Parse request file

Usage:

```
ResultParser$get_analysis_report(path = "analysis_report.csv")
```

Arguments:

`path` path to request file.

Method `get_ar()`: Parse request file

Usage:

```
ResultParser$get_ar(
  path = "ar_results.csv",
  analysis_report_tbl = NULL,
  request_tbl = NULL
)
```

Arguments:

path path to request file.
 analysis_report_tbl Parsed analysis report
 request_tbl parsed request file

Method get_car(): Parse Cumulative Abnormal Return

Usage:

```
ResultParser$get_car(path = "car_results.csv", analysis_report_tbl = NULL)
```

Arguments:

path The path to the CAR result CSV file.
 analysis_report_tbl The analysis report table. It will be used for extracting the group.

Method get_aar(): Parse AAR results

Usage:

```
ResultParser$get_aar(path = "aar_results.csv", analysis_report = NULL)
```

Arguments:

path path to aar result file.
 analysis_report Extracted analysis report

Method get_caar(): Parse caar results

Usage:

```
ResultParser$get_caar(path = "caar_results.csv")
```

Arguments:

path path to caar result file.

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
ResultParser$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

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

aar_results, [2](#)
AARResults (aar_results), [2](#)
ARCAApplicationInput, [6](#), [21](#)
ARResults (aar_results), [2](#)
AVCAApplicationInput, [9](#)
AVyCAApplicationInput, [11](#)

CAAR Results., [12](#)
CAARResults (CAAR Results.), [12](#)
CARResults (aar_results), [2](#)
checkFile, [13](#)
checkFiles, [14](#)

estAPIKey, [15](#)
EventStudy, [15](#)
EventStudy::ApplicationInputInterface,
[7](#), [10](#), [11](#), [21](#)
EventStudy::ARCAApplicationInput, [10](#)
EventStudy::EventStudyApplicationInput,
[7](#), [10](#), [11](#)
EventStudyAddin, [16](#)
EventStudyAPI, [16](#)
EventStudyApplicationInput, [21](#)

getSP500ExampleFiles, [22](#)

R6Class, [2](#), [9](#), [11](#), [13](#), [16](#), [21](#), [22](#)
ResultParser, [22](#)