

# Package ‘jsmodule’

October 11, 2021

**Title** 'RStudio' Addins and 'Shiny' Modules for Medical Research

**Version** 1.2.0

**Date** 2021-10-11

**Description**

'RStudio' addins and 'Shiny' modules for descriptive statistics, regression and survival analysis.

**Depends** R (>= 3.4.0)

**License** Apache License 2.0

**Encoding** UTF-8

**RoxygenNote** 7.1.1

**Imports** utils, stats, data.table, shiny, readxl, DT, jstable,  
labelled, methods, GGally, ggplot2, haven, rstudioapi,  
shinycustomloader, MatchIt(>= 3.0.0), survey, jskm(>= 0.4.2),  
survival, purrr, geepack, maxstat, survC1, survIDINRI, timeROC,  
shinyWidgets, pROC, Hmisc, see, readr, RColorBrewer, ggpubr,  
officer, rvg, devEMF, epiDisplay

**URL** <https://github.com/jinseob2kim/jsmodule>

**BugReports** <https://github.com/jinseob2kim/jsmodule/issues>

**Suggests** testthat, shinytest, knitr, rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Jinseob Kim [aut, cre] (<<https://orcid.org/0000-0002-9403-605X>>),  
Zarathu [cph, fnd]

**Maintainer** Jinseob Kim <jinseob2kim@gmail.com>

**Repository** CRAN

**Date/Publication** 2021-10-11 04:30:02 UTC

**R topics documented:**

coxModule	3
coxUI	5
csvFile	5
csvFileInput	6
FilePs	7
FilePsInput	9
FileRepeated	10
FileRepeatedInput	11
FileSurvey	12
FileSurveyInput	13
GEEModuleLinear	14
GEEModuleLogistic	16
GEEModuleUI	17
ggpairsModule	19
ggpairsModule2	20
ggpairsModuleUI1	21
ggpairsModuleUI2	23
ggplotdownUI	24
jsBasicAddin	25
jsBasicExtAddin	25
jsBasicGadget	26
jsPropensityAddin	27
jsPropensityExtAddin	27
jsPropensityGadget	28
jsRepeatedAddin	29
jsRepeatedExtAddin	30
jsRepeatedGadget	30
jsSurveyAddin	31
jsSurveyExtAddin	32
jsSurveyGadget	32
kaplanModule	33
kaplanUI	35
logistic.display2	36
logisticModule2	37
mklist	38
mksetdiff	39
optionUI	39
reclassificationJS	40
regress.display2	42
regressModule2	43
regressModuleUI	44
rocModule	45
rocModule2	47
rocUI	49
ROC_table	50
scatterServer	51

scatterUI . . . . .	52
survIDINRI_helper . . . . .	53
tblmodule . . . . .	54
tblmodule2 . . . . .	55
tblmoduleUI . . . . .	57
tblsimple . . . . .	58
tblsimple2 . . . . .	61
tblsimpleUI . . . . .	63
timeROChelper . . . . .	66
timerocModule . . . . .	67
timerocUI . . . . .	70
timeROC_table . . . . .	71

**Index** **72**

---

coxModule	<i>coxModule: shiny module server for Cox's model.</i>
-----------	--

---

**Description**

Shiny module server for Cox's model.

**Usage**

```
coxModule(
  input,
  output,
  session,
  data,
  data_label,
  data_varStruct = NULL,
  nfactor.limit = 10,
  design.survey = NULL,
  default.unires = T,
  limit.unires = 20,
  id.cluster = NULL,
  ties.coxph = "efron"
)
```

**Arguments**

input	input
output	output
session	session
data	reactive data
data_label	reactive data label
data_varStruct	reactive list of variable structure, Default: NULL

`nfactor.limit`    nlevels limit in factor variable, Default: 10  
`design.survey`    reactive survey data. default: NULL  
`default.unires`    Set default independent variables using univariate analysis.  
`limit.unires`    Change to `default.unires = F` if number of independent variables > `limit.unires`,  
                   Default: 20  
`id.cluster`        reactive cluster variable if marginal cox model, Default: NULL  
`ties.coxph`        'coxph' ties option, one of 'efron', 'breslow', 'exact', default: 'erfon'

### Details

Shiny modulde server for Cox's model.

### Value

Shiny modulde server for Cox's model.

### Examples

```

library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      coxUI("cox")
    ),
    mainPanel(
      DTOutput("coxtable")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_cox <- callModule(coxModule, "cox", data = data, data_label = data.label,
                       data_varStruct = NULL)

  output$coxtable <- renderDT({
    datatable(out_cox()$table, rownames=T, caption = out_cox()$caption)
  })
}

```

---

coxUI	<i>coxUI: shiny modulde UI for Cox's model.</i>
-------	---

---

**Description**

Shiny modulde UI for Cox's model.

**Usage**

```
coxUI(id)
```

**Arguments**

id	id
----	----

**Details**

Shiny modulde UI for Cox's model.

**Value**

coxUI

**Examples**

```
coxUI(1)
```

---

csvFile	<i>csvFile: Shiny module Server for file upload.</i>
---------	--

---

**Description**

Shiny module Server for file(csv or xlsx) upload.

**Usage**

```
csvFile(input, output, session, nfactor.limit = 20)
```

**Arguments**

input	input
output	output
session	session
nfactor.limit	nfactor limit to include, Default: 20

**Details**

Shiny module Server for file(csv or xlsx) upload.

**Value**

Shiny module Server for file(csv or xlsx) upload.

**Examples**

```
library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      csvFileInput("datafile")
    ),
    mainPanel(
      tabsetPanel(type = "pills",
        tabPanel("Data", DTOutput("data")),
        tabPanel("Label", DTOutput("data_label", width = "100%"))
      )
    )
  )
)

server <- function(input, output, session) {
  data <- callModule(csvFile, "datafile")

  output$data <- renderDT({
    data()$data
  })

  output$label <- renderDT({
    data()$label
  })
}
```

---

csvFileInput

*csvFileInput: Shiny module UI for file upload.*

---

**Description**

Shiny module UI for file(csv or xlsx) upload.

**Usage**

```
csvFileInput(id, label = "Upload data (csv/xlsx/sav/sas7bdat/dta)")
```

**Arguments**

id	id
label	label, Default: 'csv/xlsx/sav/sas7bdat/dta file'

**Details**

Shiny module UI for file(csv or xlsx) upload.

**Value**

Shiny module UI for file(csv or xlsx) upload.

**Examples**

```
library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      csvFileInput("datafile")
    ),
    mainPanel(
      tabsetPanel(type = "pills",
        tabPanel("Data", DTOutput("data")),
        tabPanel("Label", DTOutput("data_label", width = "100%"))
      )
    )
  )
)

server <- function(input, output, session) {
  data <- callModule(csvFile, "datafile")

  output$data <- renderDT({
    data()$data
  })

  output$label <- renderDT({
    data()$label
  })
}
```

---

FilePs

*FilePs: Shiny module Server for file upload for propensity score matching.*

---

**Description**

Shiny module Server for file upload for propensity score matching.

**Usage**

```
FilePs(input, output, session, nfactor.limit = 20)
```

**Arguments**

input	input
output	output
session	session
nfactor.limit	nfactor limit to include, Default: 20

**Details**

Shiny module Server for file upload for propensity score matching.

**Value**

Shiny module Server for file upload for propensity score matching.

**Examples**

```
library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      FilePsInput("datafile")
    ),
    mainPanel(
      tabsetPanel(type = "pills",
        tabPanel("Data", DTOutput("data")),
        tabPanel("Matching data", DTOutput("matdata")),
        tabPanel("Label", DTOutput("data_label", width = "100%"))
      )
    )
  )
)

server <- function(input, output, session) {
  mat.info <- callModule(FilePs, "datafile")

  output$data <- renderDT({
    mat.info()$data
  })

  output$matdata <- renderDT({
    mat.info()$matdata
  })

  output$label <- renderDT({
    mat.info()$label
  })
}
```



---

FilePsInput	<i>FilePsInput: Shiny module UI for file upload for propensity score matching.</i>
-------------	--

---

### Description

Shiny module UI for file upload for propensity score matching.

### Usage

```
FilePsInput(id, label = "Upload data (csv/xlsx/sav/sas7bdat/dta)")
```

### Arguments

id	id
label	label, Default: 'csv/xlsx/sav/sas7bdat file'

### Details

Shiny module UI for file upload for propensity score matching.

### Value

Shiny module UI for file upload for propensity score matching.

### Examples

```
library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      FilePsInput("datafile")
    ),
    mainPanel(
      tabsetPanel(type = "pills",
        tabPanel("Data", DTOutput("data")),
        tabPanel("Matching data", DTOutput("matdata")),
        tabPanel("Label", DTOutput("data_label", width = "100%"))
      )
    )
  )
)

server <- function(input, output, session) {
  mat.info <- callModule(FilePs, "datafile")

  output$data <- renderDT({
    mat.info()$data
  })
}
```

```

output$matdata <- renderDT({
  mat.info()$matdata
})

output$label <- renderDT({
  mat.info()$label
})
}

```

---

FileRepeated	<i>FileRepeated: File upload server module for repeated measure analysis.</i>
--------------	---

---

### Description

File upload server module for repeated measure analysis.

### Usage

```
FileRepeated(input, output, session, nfactor.limit = 20)
```

### Arguments

input	input
output	output
session	session
nfactor.limit	nfactor limit to include, Default: 20

### Details

File upload server module for repeated measure analysis.

### Value

File upload server module for repeated measure analysis.

### Examples

```

library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      FileRepeatedInput("datafile")
    ),
    mainPanel(
      tabsetPanel(type = "pills",
        tabPanel("Data", DTOutput("data")),
        tabPanel("Label", DTOutput("data_label", width = "100%"))
      )
    )
  )

```

```

    )
  )
)

server <- function(input, output, session) {
  data <- callModule(FileRepeated, "datafile")

  output$data <- renderDT({
    data()$data
  })

  output$label <- renderDT({
    data()$label
  })
}

```

---

FileRepeatedInput

*FileRepeatedInput: File upload UI for repeated measure analysis.*


---

### Description

File upload UI for repeated measure analysis.

### Usage

```
FileRepeatedInput(id, label = "Upload data (csv/xlsx/sav/sas7bdat/dta)")
```

### Arguments

id	id
label	label, Default: 'csv/xlsx/sav/sas7bdat/dta file'

### Details

File upload UI for repeated measure analysis.

### Value

File upload UI for repeated measure analysis.

### Examples

```

library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      FileRepeatedInput("datafile")
    ),

```

```

    mainPanel(
      tabsetPanel(type = "pills",
        tabPanel("Data", DTOutput("data")),
        tabPanel("Label", DTOutput("data_label", width = "100%"))
      )
    )
  )
)

server <- function(input, output, session) {
  data <- callModule(FileRepeated, "datafile")

  output$data <- renderDT({
    data()$data
  })

  output$label <- renderDT({
    data()$label
  })
}

```

---

FileSurvey

*FileSurvey: File upload server module for survey data analysis.*


---

## Description

File upload server module for survey data analysis.

## Usage

```
FileSurvey(input, output, session, nfactor.limit = 20)
```

## Arguments

input	input
output	output
session	session
nfactor.limit	nfactor limit to include, Default: 20

## Details

File upload server module for survey data analysis.

## Value

File upload server module for survey data analysis.

**Examples**

```

library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      FileSurveyInput("datafile")
    ),
    mainPanel(
      tabsetPanel(type = "pills",
        tabPanel("Data", DTOutput("data")),
        tabPanel("Label", DTOutput("data_label", width = "100%"))
      )
    )
  )
)

server <- function(input, output, session) {
  data <- callModule(FileSurvey, "datafile")

  output$data <- renderDT({
    data()$data
  })

  output$label <- renderDT({
    data()$label
  })
}

```

---

FileSurveyInput

*FileSurveyInput: File upload UI for survey data analysis.*


---

**Description**

File upload UI for survey data analysis.

**Usage**

```
FileSurveyInput(id, label = "Upload data (csv/xlsx/sav/sas7bdat/dta)")
```

**Arguments**

id	id
label	label, Default: 'csv/xlsx/sav/sas7bdat/dta file'

**Details**

File upload UI for survey data analysis.

**Value**

File upload UI for survey data analysis.

**Examples**

```
library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      FileSurveyInput("datafile")
    ),
    mainPanel(
      tabsetPanel(type = "pills",
        tabPanel("Data", DTOutput("data")),
        tabPanel("Label", DTOutput("data_label", width = "100%"))
      )
    )
  )
)

server <- function(input, output, session) {
  data <- callModule(FileSurvey, "datafile")

  output$data <- renderDT({
    data()$data
  })

  output$label <- renderDT({
    data()$label
  })
}
```

---

GEEModuleLinear

*GEEModuleLinear: shiny modulde server for gaussian generalized estimating equation(GEE) using reactive data.*

---

**Description**

Shiny modulde server for gaussian generalized estimating equation(GEE) using reactive data.

**Usage**

```
GEEModuleLinear(
  input,
  output,
  session,
  data,
  data_label,
  data_varStruct = NULL,
```

```

    nfactor.limit = 10,
    id.gee
  )

```

### Arguments

input	input
output	output
session	session
data	reactive data, ordered by id.
data_label	reactive data label
data_varStruct	List of variable structure, Default: NULL
nfactor.limit	nlevels limit in factor variable, Default: 10
id.gee	reactive repeated measure variable

### Details

Shiny modulde server for gaussian generalized estimating equation(GEE) using reactive data.

### Value

Shiny modulde server for gaussian generalized estimating equation(GEE).

### Examples

```

library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      GEEModuleUI("linear")
    ),
    mainPanel(
      DTOutput("lineartable")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))
  id.gee <- reactive("mpg")

  out_linear <- callModule(GEEModuleLinear, "linear", data = data, data_label = data.label,
    data_varStruct = NULL, id.gee = id.gee)

  output$lineartable <- renderDT({
    hide = which(colnames(out_linear())$table) == "sig")
    datatable(out_linear()$table, rownames=T, extension= "Buttons", caption = out_linear()$caption,

```

```

        options = c(opt.tbreg(out_linear())$caption),
                  list(columnDefs = list(list(visible=FALSE, targets =hide))
                    ),
                  list(scrollX = TRUE)
        )
    ) %>% formatStyle("sig", target = 'row', backgroundColor = styleEqual("**", 'yellow'))
  })
}

```

---

GEEModuleLogistic      *GEEModuleLogistic: shiny modulde server for binomial gaussian generalized estimating equation(GEE) using reactive data.*

---

### Description

Shiny modulde server for binomial gaussian generalized estimating equation(GEE) using reactive data.

### Usage

```

GEEModuleLogistic(
  input,
  output,
  session,
  data,
  data_label,
  data_varStruct = NULL,
  nfactor.limit = 10,
  id.gee
)

```

### Arguments

input	input
output	output
session	session
data	reactive data, ordered by id.
data_label	reactive data label
data_varStruct	List of variable structure, Default: NULL
nfactor.limit	nlevels limit in factor variable, Default: 10
id.gee	reactive repeated measure variable

### Details

Shiny modulde server for binomial gaussian generalized estimating equation(GEE) using reactive data.



**Value**

Shiny modulde server for binomial gaussian generalized estimating equation(GEE).

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      GEEModuleUI("logistic")
    ),
    mainPanel(
      DTOutput("logistictable")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))
  id.gee <- reactive("mpg")

  out_logistic <- callModule(GEEModuleLogistic, "logistic", data = data, data_label = data.label,
    data_varStruct = NULL, id.gee = id.gee)

  output$logistictable <- renderDT({
    hide = which(colnames(out_logistic()$table) == "sig")
    datatable(out_logistic()$table, rownames=T, extension= "Buttons",
      caption = out_logistic()$caption,
      options = c(opt.tbreg(out_logistic()$caption),
        list(columnDefs = list(list(visible=FALSE, targets =hide))
        ),
        list(scrollX = TRUE)
      )
    ) %>% formatStyle("sig", target = 'row', backgroundColor = styleEqual("**", 'yellow'))
  })
}
```

---

GEEModuleUI

*GEEModuleUI: shiny modulde UI for generalized estimating equation(GEE).*

---

**Description**

Shiny modulde UI for generalized estimating equation(GEE).

**Usage**

GEEModuleUI(id)

**Arguments**

id id

**Details**

Shiny module UI for generalized estimating equation(GEE).

**Value**

Shiny module UI for generalized estimating equation(GEE).

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      GEEModuleUI("linear")
    ),
    mainPanel(
      DTOutput("lineartable")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))
  id.gee <- reactive("mpg")

  out_linear <- callModule(GEEModuleLinear, "linear", data = data, data_label = data.label,
    data_varStruct = NULL, id.gee = id.gee)

  output$lineartable <- renderDT({
    hide = which(colnames(out_linear())$table) == "sig")
    datatable(out_linear())$table, rownames=T, extension= "Buttons", caption = out_linear()$caption,
    options = c(opt.tbreg(out_linear())$caption,
      list(columnDefs = list(list(visible=FALSE, targets =hide))
    ),
    list(scrollX = TRUE)
  )
  ) %>% formatStyle("sig", target = 'row',backgroundColor = styleEqual("**", 'yellow'))
})
}
```

---

`ggpairsModule`*ggpairsModule: shiny module server for basic/scatter plot.*

---

## Description

Shiny module server for basic/scatter plot.

## Usage

```
ggpairsModule(  
  input,  
  output,  
  session,  
  data,  
  data_label,  
  data_varStruct = NULL,  
  nfactor.limit = 20  
)
```

## Arguments

<code>input</code>	<code>input</code>
<code>output</code>	<code>output</code>
<code>session</code>	<code>session</code>
<code>data</code>	<code>data</code>
<code>data_label</code>	data label
<code>data_varStruct</code>	List of variable structure, Default: NULL
<code>nfactor.limit</code>	nlevels limit for categorical variables, Default: 20

## Details

Shiny module server for basic/scatter plot.

## Value

Shiny module server for basic/scatter plot.

## Examples

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)  
library(GGally)  
  
ui <- fluidPage(  
  sidebarLayout(  
    sidebarPanel(  
      ggpairsModuleUI1("ggpairs")
```

```

    ),
    mainPanel(
      plotOutput("ggpairs_plot"),
      ggpairsModuleUI2("ggpairs")
    )
  )
)

server <- function(input, output, session) {

  data <- mtcars
  data.label <- jstable::mk.lev(mtcars)

  out_ggpairs <- callModule(ggpairsModule, "ggpairs", data = data, data_label = data.label,
                           data_varStruct = NULL)

  output$kaplan_plot <- renderPlot({
    print(out_ggpairs())
  })
}

```

---

ggpairsModule2

*ggpairsModule2: shiny module server for basic/scatter plot for reactive data.*


---

## Description

Shiny module server for basic/scatter plot for reactive data.

## Usage

```

ggpairsModule2(
  input,
  output,
  session,
  data,
  data_label,
  data_varStruct = NULL,
  nfactor.limit = 20
)

```

## Arguments

input	input
output	output
session	session
data	Reactive data
data_label	Reactive data label

`data_varStruct` List of variable structure, Default: NULL  
`nfactor.limit` nlevels limit for categorical variables, Default: 20

### Details

Shiny module server for basic/scatter plot for reactive data.

### Value

Shiny module server for basic/scatter plot

### Examples

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
library(GGally)

ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      ggpairsModuleUI1("ggpairs")
    ),
    mainPanel(
      plotOutput("ggpairs_plot"),
      ggpairsModuleUI2("ggpairs")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_ggpairs <- callModule(ggpairsModule2, "ggpairs", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$kaplan_plot <- renderPlot({
    print(out_ggpairs())
  })
}
```

---

ggpairsModuleUI1

*ggpairsModuleUI1: Variable selection module UI for ggpairs*

---

### Description

Variable selection module UI for ggpairs

**Usage**

```
ggpairsModuleUI1(id)
```

**Arguments**

```
id          id
```

**Details**

Variable selection module UI for ggpairs

**Value**

Variable selection module UI for ggpairs

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
library(GGally)

ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      ggpairsModuleUI1("ggpairs")
    ),
    mainPanel(
      plotOutput("ggpairs_plot"),
      ggpairsModuleUI2("ggpairs")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_ggpairs <- callModule(ggpairsModule2, "ggpairs", data = data, data_label = data.label,
                           data_varStruct = NULL)

  output$kaplan_plot <- renderPlot({
    print(out_ggpairs())
  })
}
```

---

ggpairsModuleUI2      *ggpairsModuleUI2: Option & download module UI for ggpairs*

---

**Description**

Option & download module UI for ggpairs

**Usage**

```
ggpairsModuleUI2(id)
```

**Arguments**

id                    id

**Details**

Option & download module UI for ggpairs

**Value**

Option & download module UI for ggpairs

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
library(GGally)

ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      ggpairsModuleUI1("ggpairs")
    ),
    mainPanel(
      plotOutput("ggpairs_plot"),
      ggpairsModuleUI2("ggpairs")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_ggpairs <- callModule(ggpairsModule2, "ggpairs", data = data, data_label = data.label,
                           data_varStruct = NULL)

  output$kaplan_plot <- renderPlot({
    print(out_ggpairs())
  })
}
```

```
  })
}
```

---

ggplotdownUI

*ggplotdownUI: Option & download module UI for ggplot*


---

## Description

Option & download module UI for ggplot

## Usage

```
ggplotdownUI(id)
```

## Arguments

```
id          id
```

## Details

Option & download module UI for ggplot

## Value

Option & download module UI for ggplot

## Examples

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      kaplanUI("kaplan")
    ),
    mainPanel(
      plotOutput("kaplan_plot"),
      ggplotdownUI("kaplan")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_kaplan <- callModule(kaplanModule, "kaplan", data = data, data_label = data.label,
                           data_varStruct = NULL)

  output$kaplan_plot <- renderPlot({
```



```
    print(out_kaplan())
  })
}
```

---

jsBasicAddin

*jsBasicAddin: Rstudio addin of jsBasicGadget*

---

### Description

Rstudio addin of jsBasicGadget

### Usage

```
jsBasicAddin()
```

### Details

Rstudio addin of jsBasicGadget

### Value

Rstudio addin of jsBasicGadget

### See Also

[rstudio-editors](#)

### Examples

```
if(interactive()){
  jsBasicAddin()
}
```

---

jsBasicExtAddin

*jsBasicExtAddin: RStudio Addin for basic data analysis with external data.*

---

### Description

RStudio Addin for basic data analysis with external csv/xlsx/sas7bdat/sav/dta file.

### Usage

```
jsBasicExtAddin(nfactor.limit = 20, max.filesize = 2048)
```

**Arguments**

`nfactor.limit` nlevels limit for categorical variables, Default: 20  
`max.filesize` Maximum file size to upload (MB), Default: 2048 (2 GB)

**Details**

RStudio Addin for basic data analysis with external csv/xlsx/sas7bdat/sav/dta file.

**Value**

RStudio Addin for basic data analysis with external data.

**See Also**

[lung fwrite opt. tbreg](#)

**Examples**

```
if(interactive()){
  jsBasicExtAddin()
}
```

---

jsBasicGadget

*jsBasicGadget: Shiny Gadget of Basic Statistics in Medical Research.*

---

**Description**

Shiny Gadget including Data, Label info, Table 1, Regression(linear, logistic), Basic plot

**Usage**

```
jsBasicGadget(data, nfactor.limit = 20)
```

**Arguments**

`data` data  
`nfactor.limit` nlevels limit for categorical variables

**Details**

Shiny Gadget including Data, Label info, Table 1, Regression(linear, logistic), Basic plot

**Value**

Shiny Gadget including Data, Label info, Table 1, Regression(linear, logistic), Basic plot

**Examples**

```
if(interactive()){  
  jsBasicGadget(mtcars)  
}
```

---

jsPropensityAddin      *jsPropensityAddin: Rstudio addin of jsPropensityGadget*

---

**Description**

Rstudio addin of jsPropensityGadget

**Usage**

```
jsPropensityAddin()
```

**Details**

Rstudio addin of jsPropensityGadget

**Value**

Rstudio addin of jsPropensityGadget

**See Also**

[rstudio-editors](#)

**Examples**

```
if(interactive()){  
  jsPropensityAddin()  
}
```

---

jsPropensityExtAddin      *jsPropensityExtAddin: RStudio Addin for propensity score analysis with external data.*

---

**Description**

RStudio Addin for propensity score analysis with external csv/xlsx/sas7bdat/sav/dta file.

**Usage**

```
jsPropensityExtAddin(nfactor.limit = 20, max.filesize = 2048)
```

**Arguments**

`nfactor.limit` nlevels limit for categorical variables, Default: 20  
`max.filesize` Maximum file size to upload (MB), Default: 2048 (2 GB)

**Details**

RStudio Addin for propensity score analysis with external csv/xlsx/sas7bdat/sav/dta file.

**Value**

RStudio Addin for propensity score analysis with external data.

**See Also**

[pbc.fwrite](#), [data.table](#), [svydesign](#), [opt.tbreg](#)

**Examples**

```
if(interactive()){
  jsPropensityExtAddin()
}
```

---

`jsPropensityGadget`     *jsPropensityGadget: Shiny Gadget for propensity score analysis.*

---

**Description**

Shiny Gadget including original/matching/IPTW data, Label info, Table 1, Cox model, Basic/kaplan-meier plot.

**Usage**

```
jsPropensityGadget(data, nfactor.limit = 20)
```

**Arguments**

`data` data  
`nfactor.limit` nlevels limit for categorical variables, Default: 20

**Details**

Shiny Gadget including original/matching/IPTW data, Label info, Table 1, Cox model, Basic/kaplan-meier plot.

**Value**

Shiny Gadget including original/matching/IPTW data, Label info, Table 1, Cox model, Basic/kaplan-meier plot.

**See Also**

[data.table](#), [matchit](#), [match.data](#), [cox2.display](#), [svycox.display](#), [survfit](#), [coxph](#), [Surv](#), [jskm](#), [svyjskm](#), [ggsave](#), [svykm](#)

**Examples**

```
if(interactive()){
  jsPropensityGadget(mtcars)
}
```

---

jsRepeatedAddin

*jsRepeatedAddin: Rstudio addin of jsRepeatedGadget*

---

**Description**

Rstudio addin of jsRepeatedGadget

**Usage**

```
jsRepeatedAddin()
```

**Details**

Rstudio addin of jsRepeatedGadget

**Value**

Rstudio addin of jsRepeatedGadget

**See Also**

[rstudio-editors](#)

**Examples**

```
if(interactive()){
  jsRepeatedAddin()
}
```

jsRepeatedExtAddin     *jsRepeatedExtAddin: RStudio Addin for repeated measure analysis with external data.*

---

### Description

RStudio Addin for repeated measure analysis with external csv/xlsx/sas7bdat/sav/dta file.

### Usage

```
jsRepeatedExtAddin(nfactor.limit = 20, max.filesize = 2048)
```

### Arguments

`nfactor.limit`    nlevels limit for categorical variables, Default: 20  
`max.filesize`    Maximum file size to upload (MB), Default: 2048 (2 GB)

### Details

RStudio Addin for repeated measure analysis with external csv/xlsx/sas7bdat/sav/dta file.

### Value

RStudio Addin for repeated measure analysis with external data.

### See Also

[fwrite](#) [colon](#) [opt.tbreg](#)

### Examples

```
if(interactive()){  
  jsRepeatedExtAddin()  
}
```

---

jsRepeatedGadget     *jsRepeatedGadget: Shiny Gadget of Repeated measure analysis.*

---

### Description

Shiny Gadget including Data, Label info, Table 1, GEE(linear, logistic), Basic plot

### Usage

```
jsRepeatedGadget(data, nfactor.limit = 20)
```

**Arguments**

data                    data  
nfactor.limit    nlevels limit for categorical variables

**Details**

Shiny Gadget including Data, Label info, Table 1, GEE(linear, logistic), Basic plot

**Value**

Shiny Gadget including Data, Label info, Table 1, GEE(linear, logistic), Basic plot

**Examples**

```
if(interactive()){  
  jsRepeatedGadget(mtcars)  
}
```

---

jsSurveyAddin

*jsSurveyAddin: Rstudio addin of jsSurveyGadget*

---

**Description**

Rstudio addin of jsSurveyGadget

**Usage**

```
jsSurveyAddin()
```

**Details**

Rstudio addin of jsSurveyGadget

**Value**

Rstudio addin of jsSurveyGadget

**See Also**

[rstudio-editors](#)

**Examples**

```
if(interactive()){  
  jsSurveydAddin()  
}
```

---

jsSurveyExtAddin	<i>jsSurveyExtAddin: RStudio Addin for survey data analysis with external data.</i>
------------------	---

---

**Description**

RStudio Addin for survey data analysis with external csv/xlsx/sas7bdat/sav/dta file.

**Usage**

```
jsSurveyExtAddin(nfactor.limit = 20, max.filesize = 2048)
```

**Arguments**

`nfactor.limit` nlevels limit for categorical variables, Default: 20  
`max.filesize` Maximum file size to upload (MB), Default: 2048 (2 GB)

**Details**

RStudio Addin for survey data analysis with external csv/xlsx/sas7bdat/sav/dta file.

**Value**

RStudio Addin for survey data analysis with external data.

**See Also**

[fwrite](#) [opt.tb1](#) [opt.tbreg](#)

**Examples**

```
if(interactive()){
  jsSurveyExtAddin()
}
```

---

jsSurveyGadget	<i>jsSurveyGadget: Shiny Gadget of survey data analysis.</i>
----------------	--

---

**Description**

Shiny Gadget including Data, Label info, Table 1, svyglm, Basic plot

**Usage**

```
jsSurveyGadget(data, nfactor.limit = 20)
```



**Arguments**

data            data  
nfactor.limit   nlevels limit for categorical variables

**Details**

Shiny Gadget including Data, Label info, Table 1, svyglm, Basic plot

**Value**

Shiny Gadget including Data, Label info, Table 1, svyglm, Basic plot

**Examples**

```
if(interactive()){  
  jsSurveyGadget(mtcars)  
}
```

---

kaplanModule

*kaplanModule: shiny module server for kaplan-meier plot.*

---

**Description**

Shiny module server for kaplan-meier plot.

**Usage**

```
kaplanModule(  
  input,  
  output,  
  session,  
  data,  
  data_label,  
  data_varStruct = NULL,  
  nfactor.limit = 10,  
  design.survey = NULL,  
  id.cluster = NULL,  
  timeby = NULL,  
  range.x = NULL,  
  range.y = NULL  
)
```

**Arguments**

input	input
output	output
session	session
data	Reactive data
data_label	Reactive data label
data_varStruct	Reactive List of variable structure, Default: NULL
nfactor.limit	nlevels limit in factor variable, Default: 10
design.survey	Reactive survey data. default: NULL
id.cluster	Reactive cluster variable if marginal model, Default: NULL
timeby	timeby, Default: NULL
range.x	range of x axis, Default: NULL
range.y	range of y axis, Default: NULL

**Details**

Shiny module server for kaplan-meier plot.

**Value**

Shiny module server for kaplan-meier plot.

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      kaplanUI("kaplan")
    ),
    mainPanel(
      plotOutput("kaplan_plot"),
      ggplotdownUI("kaplan")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_kaplan <- callModule(kaplanModule, "kaplan", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$kaplan_plot <- renderPlot({
    print(out_kaplan())
  })
}
```

---

kaplanUI	<i>kaplanUI: shiny module UI for kaplan-meier plot</i>
----------	--

---

**Description**

Shiny module UI for kaplan-meier plot

**Usage**

```
kaplanUI(id)
```

**Arguments**

id	id
----	----

**Details**

Shiny module UI for kaplan-meier plot

**Value**

Shiny module UI for kaplan-meier plot

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      kaplanUI("kaplan")
    ),
    mainPanel(
      plotOutput("kaplan_plot"),
      ggplotdownUI("kaplan")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_kaplan <- callModule(kaplanModule, "kaplan", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$kaplan_plot <- renderPlot({
    print(out_kaplan())
  })
}
```

---

logistic.display2      *logistic.display2: Modified epiDisplay's logistic.display function.*

---

### Description

Modified epiDisplay's logistic.display function for reactive data.

### Usage

```
logistic.display2(  
  logistic.model,  
  alpha = 0.05,  
  crude = TRUE,  
  crude.p.value = FALSE,  
  decimal = 2,  
  simplified = FALSE  
)
```

### Arguments

logistic.model	glm object(binomial)
alpha	alpha, Default: 0.05
crude	crude, Default: TRUE
crude.p.value	crude.p.value, Default: FALSE
decimal	decimal, Default: 2
simplified	simplified, Default: FALSE

### Details

Modified epiDisplay's logistic.display function for reactive data.

### Value

logistic table

### Examples

```
model1 <- glm(am ~ cyl + disp, data = mtcars, family = binomial)  
logistic.display2(model1, crude = TRUE, crude.p.value = TRUE, decimal = 3)
```

---

logisticModule2	<i>logisticModule2: Shiny modulde server for logistic regression for reactive data.</i>
-----------------	---

---

### Description

Shiny modulde server for logistic regression for reactive data.

### Usage

```
logisticModule2(  
  input,  
  output,  
  session,  
  data,  
  data_label,  
  data_varStruct = NULL,  
  nfactor.limit = 10,  
  design.survey = NULL,  
  default.unires = T,  
  limit.unires = 20  
)
```

### Arguments

input	input
output	output
session	session
data	reactive data
data_label	reactive data label
data_varStruct	List of variable structure, Default: NULL
nfactor.limit	nlevels limit in factor variable, Default: 10
design.survey	reactive survey data. default: NULL
default.unires	Set default independent variables using univariate analysis, Default: T
limit.unires	Change to default.unires = F if number of independent variables > limit.unires, Default: 20

### Details

Shiny modulde server for logistic regression.

### Value

Shiny modulde server for logistic regression.

## Examples

```
library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      regressModuleUI("logistic")
    ),
    mainPanel(
      DTOutput("logistictable")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_logistic <- callModule(logisticModule2, "logistic", data = data, data_label = data.label,
                             data_varStruct = NULL)

  output$logistictable <- renderDT({
    datatable(out_logistic()$table, rownames=T, caption = out_logistic()$caption)
  })
}
```

---

mklist

*mklist: function to make variable list Including specific variables.*

---

## Description

Function to make variable list Including specific variables.

## Usage

```
mklist(varlist, vars)
```

## Arguments

varlist	Original variable list.
vars	variable to include.

## Details

Internal function

## Value

variable list Including specific variables.

**Examples**

```
data_varStruct <- list(variable = names(mtcars))
mklist(data_varStruct, names(mtcars))
```

---

mksetdiff	<i>mksetdiff: function to make variable list excluding specific variables.</i>
-----------	--

---

**Description**

Function to make variable list excluding specific variables.

**Usage**

```
mksetdiff(varlist, vars)
```

**Arguments**

varlist	Original variable list
vars	variable to exclude.

**Details**

Internal function

**Value**

variable list excluding specific variables.

**Examples**

```
data_varStruct <- list(variable = names(mtcars))
mksetdiff(data_varStruct, "mpg")
```

---

optionUI	<i>optionUI: Option UI with icon</i>
----------	--------------------------------------

---

**Description**

Option UI with icon

**Usage**

```
optionUI(id)
```

**Arguments**

id	id
----	----

**Details**

Option UI with icon

**Value**

Option UI with icon

**See Also**

[dropdownButton](#), [tooltipOptions](#)

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      kaplanUI("kaplan")
    ),
    mainPanel(
      optionUI("kaplan"),
      plotOutput("kaplan_plot"),
      ggplotdownUI("kaplan")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_kaplan <- callModule(kaplanModule, "kaplan", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$kaplan_plot <- renderPlot({
    print(out_kaplan())
  })
}
```

---

reclassificationJS

*reclassificationJS: Function for reclassification table and statistics*

---

**Description**

Modified function of PredictABEL::reclassification: return output table



**Usage**

```
reclassificationJS(
  data,
  cOutcome,
  predrisk1,
  predrisk2,
  cutoff,
  dec.value = 3,
  dec.p = 3
)
```

**Arguments**

<code>data</code>	Data frame or matrix that includes the outcome and predictors variables.
<code>cOutcome</code>	Column number of the outcome variable.
<code>predrisk1</code>	Vector of predicted risks of all individuals using initial model.
<code>predrisk2</code>	Vector of predicted risks of all individuals using updated model.
<code>cutoff</code>	Cutoff values for risk categories. Define the cut-off values. Ex: <code>c(0,.20,.30,1)</code>
<code>dec.value</code>	digits of value, Default: 4
<code>dec.p</code>	digits of p, Default: 3

**Details**

Modified function of `PredictABEL::reclassification`

**Value**

Table including NRI(categorical), NRI(continuous), IDI with 95

**See Also**

[rcorrp.cens](#)

**Examples**

```
m1 <- glm(vs ~ am + gear, data = mtcars, family = binomial)
m2 <- glm(vs ~ am + gear + wt, data = mtcars, family = binomial)
reclassificationJS(data = mtcars, cOutcome = 8,
  predrisk1 = predict(m1, type = "response"),
  predrisk2=predict(m2, type = "response"), cutoff = c(0, .20, .40,1))
```

---

regress.display2      *regress.display2: modified epiDisplay's regress.display function*

---

### Description

regress.display function for reactive data

### Usage

```
regress.display2(  
  regress.model,  
  alpha = 0.05,  
  crude = FALSE,  
  crude.p.value = FALSE,  
  decimal = 2,  
  simplified = FALSE  
)
```

### Arguments

regress.model	lm object
alpha	alpha, Default: 0.05
crude	crude, Default: FALSE
crude.p.value	crude.p.value, Default: FALSE
decimal	decimal, Default: 2
simplified	simplified, Default: FALSE

### Details

regress.display function for reactive data

### Value

regress table

### Examples

```
model1 <- glm(mpg ~ cyl + disp + vs, data = mtcars)  
regress.display2(model1, crude = TRUE, crude.p.value = TRUE, decimal = 3)
```

---

regressModule2	<i>regressModule2: Shiny modulde server for linear regression for reactive data.</i>
----------------	--

---

### Description

Shiny modulde server for linear regression for reactive data.

### Usage

```
regressModule2(
  input,
  output,
  session,
  data,
  data_label,
  data_varStruct = NULL,
  nfactor.limit = 10,
  design.survey = NULL,
  default.unires = T,
  limit.unires = 20
)
```

### Arguments

input	input
output	output
session	session
data	reactive data
data_label	reactive data label
data_varStruct	List of variable structure, Default: NULL
nfactor.limit	nlevels limit in factor variable, Default: 10
design.survey	reactive survey data. default: NULL
default.unires	Set default independent variables using univariate analysis, Default: T
limit.unires	Change to default.unires = F if number of independent variables > limit.unires, Default: 20

### Details

Shiny modulde server for linear regression.

### Value

Shiny modulde server for linear regression.

**Examples**

```

library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      regressModuleUI("linear")
    ),
    mainPanel(
      DTOutput("lineartable")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_linear <- callModule(regressModule2, "linear", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$lineartable <- renderDT({
    datatable(out_linear()$table, rownames=T, caption = out_linear()$caption)
  })
}

```

regressModuleUI

*regressModuleUI: shiny modulde UI for linear regression.***Description**

Shiny modulde UI for linear regression.

**Usage**

```
regressModuleUI(id)
```

**Arguments**

```
id          id
```

**Details**

Shiny modulde UI for linear regression.

**Value**

Shiny modulde UI for linear regression.

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      regressModuleUI("linear")
    ),
    mainPanel(
      DTOutput("lineartable")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_linear <- callModule(regressModule2, "linear", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$lineartable <- renderDT({
    datatable(out_linear())$table, rownames=T, caption = out_linear()$caption)
  })
}
```

---

rocModule

*rocModule: shiny module server for roc analysis*

---

**Description**

shiny module server for roc analysis

**Usage**

```
rocModule(
  input,
  output,
  session,
  data,
  data_label,
  data_varStruct = NULL,
  nfactor.limit = 10,
  design.survey = NULL,
  id.cluster = NULL
)
```

**Arguments**

input	input
output	output
session	session
data	Reactive data
data_label	Reactive data label
data_varStruct	Reactive List of variable structure, Default: NULL
nfactor.limit	nlevels limit in factor variable, Default: 10
design.survey	Reactive survey data. default: NULL
id.cluster	Reactive cluster variable if marginal model, Default: NULL

**Details**

shiny module server for roc analysis

**Value**

shiny module server for roc analysis

**See Also**

[quantile](#) [setkey](#) [ggroc](#) [geeglm](#) [svyglm](#) [theme\\_modern](#)

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2);library(pROC)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      rocUI("roc")
    ),
    mainPanel(
      plotOutput("plot_roc"),
      ggplotdownUI("roc"),
      DTOutput("table_roc")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- jstable::mk.lev(mtcars)

  out_roc <- callModule(rocModule, "roc", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$plot_roc <- renderPlot({
```

```

    print(out_roc())$plot)
  })

  output$table_roc <- renderDT({
    datatable(out_roc())$tb, rownames=F, editable = F, extensions= "Buttons",
      caption = "ROC results",
      options = c(jstable::opt.tbreg("roctable"), list(scrollX = TRUE)))
  })
}

```

---

rocModule2	<i>rocModule2: shiny module server for roc analysis- input number of model as integer</i>
------------	---

---

## Description

shiny module server for roc analysis- input number of model as integer

## Usage

```

rocModule2(
  input,
  output,
  session,
  data,
  data_label,
  data_varStruct = NULL,
  nfactor.limit = 10,
  design.survey = NULL,
  id.cluster = NULL
)

```

## Arguments

input	input
output	output
session	session
data	Reactive data
data_label	Reactive data label
data_varStruct	Reactive List of variable structure, Default: NULL
nfactor.limit	nlevels limit in factor variable, Default: 10
design.survey	Reactive survey data. default: NULL
id.cluster	Reactive cluster variable if marginal model, Default: NULL

**Details**

shiny module server for roc analysis- input number of model as integer

**Value**

shiny module server for roc analysis- input number of model as integer

**See Also**

[quantile](#) [setkey](#) [ggroc](#) [geeglm](#) [svyglm](#) [theme\\_modern](#)

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2);library(pROC)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      rocUI("roc")
    ),
    mainPanel(
      plotOutput("plot_roc"),
      ggplotdownUI("roc"),
      DTOutput("table_roc")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- jstable::mk.lev(mtcars)

  out_roc <- callModule(rocModule2, "roc", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$plot_roc <- renderPlot({
    print(out_roc())$plot
  })

  output$table_roc <- renderDT({
    datatable(out_roc())$tb, rownames=F, editable = F, extensions= "Buttons",
      caption = "ROC results",
      options = c(jstable::opt.tbreg("roctable"), list(scrollX = TRUE))
  })
}
```



---

rocUI	<i>rocUI: shiny module UI for roc analysis</i>
-------	--

---

**Description**

Shiny module UI for roc analysis

**Usage**

```
rocUI(id)
```

**Arguments**

id	id
----	----

**Details**

Shiny module UI for roc analysis

**Value**

Shiny module UI for roc analysis

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2);library(pROC)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      rocUI("roc")
    ),
    mainPanel(
      plotOutput("plot_roc"),
      ggplotdownUI("roc"),
      DTOutput("table_roc")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- jstable::mk.lev(mtcars)

  out_roc <- callModule(rocModule, "roc", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$plot_roc <- renderPlot({
    print(out_roc())$plot
  })
}
```

```

output$table_roc <- renderDT({
  datatable(out_roc())$tb, rownames=F, editable = F, extensions= "Buttons",
            caption = "ROC results",
            options = c(jstable::opt.tbreg("roctable"), list(scrollX = TRUE)))
})
}

```

---

ROC_table	<i>ROC_table: extract AUC, NRI and IDI information from list of roc object in pROC packages.</i>
-----------	--

---

### Description

extract AUC, NRI and IDI information from list of roc in pROC packages

### Usage

```
ROC_table(ListModel, dec.auc = 3, dec.p = 3)
```

### Arguments

ListModel	list of roc object
dec.auc	digits for AUC, Default: 3
dec.p	digits for p value, Default: 3

### Details

extract AUC, NRI and IDI information from list of roc object in pROC packages.

### Value

table of AUC, NRI and IDI information

### See Also

[ci.auc,roc.test.data.table](#), [rbindlist](#)

### Examples

```

library(pROC)
m1 <- glm(vs ~ am + gear, data = mtcars, family = binomial)
m2 <- glm(vs ~ am + gear + wt, data = mtcars, family = binomial)
m3 <- glm(vs ~ am + gear + wt + mpg, data = mtcars, family = binomial)
roc1 <- roc(m1$y, predict(m1, type = "response"))
roc2 <- roc(m2$y, predict(m2, type = "response"))
roc3 <- roc(m3$y, predict(m3, type = "response"))
list.roc <- list(roc1, roc2, roc3)
ROC_table(list.roc)

```

---

scatterServer	<i>scatterServer: shiny module server for scatterplot.</i>
---------------	--

---

### Description

Shiny module server for scatterplot.

### Usage

```
scatterServer(id, data, data_label, data_varStruct = NULL, nfactor.limit = 10)
```

### Arguments

id	id
data	Reactive data
data_label	Reactive data label
data_varStruct	Reactive List of variable structure, Default: NULL
nfactor.limit	nlevels limit in factor variable, Default: 10

### Details

Shiny module server for scatterplot.

### Value

Shiny module server for scatterplot.

### Examples

```
library(shiny);library(ggplot2);library(ggpubr);
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      scatterUI("scatter")
    ),
    mainPanel(
      plotOutput("scatter_plot"),
      ggplotdownUI("scatter")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_scatter <- scatterServer("scatter", data = data, data_label = data.label,
```

```
    data_varStruct = NULL)

    output$scatter_plot <- renderPlot({
      print(out_scatter())
    })
  }
}
```

---

scatterUI

*scatterUI: shiny module UI for scatterplot*

---

### Description

Shiny module UI for scatterplot

### Usage

```
scatterUI(id, label = "scatterplot")
```

### Arguments

id	id
label	label

### Details

Shiny module UI for scatterplot

### Value

Shiny module UI for scatterplot

### Examples

```
library(shiny);library(ggplot2);library(ggpubr);
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      scatterUI("scatter")
    ),
    mainPanel(
      plotOutput("scatter_plot"),
      ggplotdownUI("scatter")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
```

```

data.label <- reactive(jstable::mk.lev(mtcars))

out_scatter <- scatterServer("scatter", data = data, data_label = data.label,
  data_varStruct = NULL)

output$scatter_plot <- renderPlot({
  print(out_scatter())
})
}

```

---

survIDINRI\_helper      *survIDINRI\_helper: Helper function for IDI.INF.OUT in survIDINRI packages*

---

## Description

Helper function for IDI.INF.OUT in survIDINRI packages

## Usage

```

survIDINRI_helper(
  var.event,
  var.time,
  list.vars.ind,
  t,
  data,
  dec.auc = 3,
  dec.p = 3,
  id.cluster = NULL
)

```

## Arguments

var.event	event
var.time	time
list.vars.ind	list of independent variable
t	time
data	data
dec.auc	digits for AUC, Default: 3
dec.p	digits for p value, Default: 3
id.cluster	cluster variable if marginal model, Default: NULL

## Details

Helper function for IDI.INF.OUT in survIDINRI packages

**Value**

IDI, NRI

**See Also**

[data.table model.matrix coxph Surv IDI.INF.OUT IDI.INF](#)

**Examples**

```
#library(survival)
#survIDINRI_helper("status", "time", list.vars.ind = list("age", c("age", "sex")),
#                  t = 365, data = lung)
```

---

tb1module

*tb1module: table 1 shiny module server.*

---

**Description**

Table 1 shiny module server for descriptive statistics.

**Usage**

```
tb1module(
  input,
  output,
  session,
  data,
  data_label,
  data_varStruct = NULL,
  nfactor.limit = 10,
  design.survey = NULL,
  showAllLevels = T,
  argsExact = list(workspace = 2 * 10^7, simulate.p.value = T)
)
```

**Arguments**

input	input
output	output
session	session
data	Data
data_label	Data label
data_varStruct	Variable structure list of data, Default: NULL
nfactor.limit	maximum factor levels to include, Default: 10
design.survey	survey data of survey package. default: NULL
showAllLevels	Show All label information with 2 categorical variables, Default: T
argsExact	Option for Fisher exact test memory limit.

**Details**

Table 1 shiny module server for descriptive statistics.

**Value**

Table 1 shiny module server for descriptive statistics.

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      tb1moduleUI("tb1")
    ),
    mainPanel(
      DTOutput("table1")
    )
  )
)

server <- function(input, output, session) {

  data <- mtcars
  data.label <- jstable::mk.lev(mtcars)

  out_tb1 <- callModule(tb1module, "tb1", data = data, data_label = data.label,
    data_varStruct = NULL)

  output$table1 <- renderDT({
    tb <- out_tb1()$table
    cap <- out_tb1()$caption
    out.tb1 <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
    return(out.tb1)
  })
}
```

---

 tb1module2

*tb1module2: table 1 shiny module server for reactive data.*


---

**Description**

Table 1 shiny module server for descriptive statistics for reactive data.

**Usage**

```
tb1module2(
  input,
  output,
```

```

    session,
    data,
    data_label,
    data_varStruct = NULL,
    nfactor.limit = 10,
    design.survey = NULL,
    showAllLevels = T,
    argsExact = list(workspace = 2 * 10^7, simulate.p.value = T)
  )

```

### Arguments

input	input
output	output
session	session
data	Reactive data
data_label	Reactive data label
data_varStruct	Variable structure list of data, Default: NULL
nfactor.limit	maximum factor levels to include, Default: 10
design.survey	Reactive survey data of survey package. Default: NULL
showAllLevels	Show All label information with 2 categorical variables, Default: T
argsExact	Option for Fisher exact test memory limit.

### Details

Table 1 shiny module server for descriptive statistics.

### Value

Table 1 shiny module server for descriptive statistics.

### Examples

```

library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      tb1moduleUI("tb1")
    ),
    mainPanel(
      DTOutput("table1")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)

```



```

data.label <- reactive(jstable::mk.lev(mtcars))

out_tb1 <- callModule(tb1module2, "tb1", data = data, data_label = data.label,
                    data_varStruct = NULL)

output$table1 <- renderDT({
  tb <- out_tb1()$table
  cap <- out_tb1()$caption
  out.tb1 <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
  return(out.tb1)
})
}

```

---

tb1moduleUI

*tb1moduleUI: table 1 module UI.*


---

### Description

Table 1 shiny module UI for descriptive statistics.

### Usage

```
tb1moduleUI(id)
```

### Arguments

```
id          id
```

### Details

Table 1 shiny module UI for descriptive statistics.

### Value

Table 1 module UI.

### Examples

```

library(shiny);library(DT);library(data.table);library(jstable)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      tb1moduleUI("tb1")
    ),
    mainPanel(
      DTOutput("table1")
    )
  )
)
)

```

```

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- reactive(jstable::mk.lev(mtcars))

  out_tb1 <- callModule(tb1module2, "tb1", data = data, data_label = data.label,
                       data_varStruct = NULL)

  output$table1 <- renderDT({
    tb <- out_tb1()$table
    cap <- out_tb1()$caption
    out.tb1 <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
    return(out.tb1)
  })
}

```

---

 tb1simple

*tb1simple: tb1 module server for propensity score analysis*


---

## Description

Table 1 module server for propensity score analysis

## Usage

```

tb1simple(
  input,
  output,
  session,
  data,
  matdata,
  data_label,
  data_varStruct = NULL,
  group_var,
  showAllLevels = T
)

```

## Arguments

input	input
output	output
session	session
data	Original data with propensity score
matdata	Matching data
data_label	Data label
data_varStruct	List of variable structure, Default: NULL
group_var	Group variable to run propensity score analysis.
showAllLevels	Show All label information with 2 categorical variables, Default: T

**Details**

Table 1 module server for propensity score analysis

**Value**

Table 1 with original data/matching data/IPTW data

**See Also**

[var\\_label CreateTableOneJS svydesign](#)

**Examples**

```
library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
library(haven);library(survey)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      FilePsInput("datafile"),
      tb1simpleUI("tb1")
    ),
    mainPanel(
      DTOutput("table1_original"),
      DTOutput("table1_ps"),
      DTOutput("table1_iptw")
    )
  )
)

server <- function(input, output, session) {

  mat.info <- callModule(FilePs, "datafile")

  data <- reactive(mat.info())$data
  matdata <- reactive(mat.info())$matdata
  data.label <- reactive(mat.info())$data.label

  vlist <- eventReactive(mat.info(), {
    mklist <- function(varlist, vars){
      lapply(varlist,
        function(x){
          inter <- intersect(x, vars)
          if (length(inter) == 1){
            inter <- c(inter, "")
          }
          return(inter)
        })
    }
  })
  factor_vars <- names(data())[data()[, lapply(.SD, class) %in% c("factor", "character")]]
  factor_list <- mklist(data_varStruct(), factor_vars)
  conti_vars <- setdiff(names(data()), c(factor_vars, "pscore", "iptw"))
```

```

  conti_list <- mklst(data_varStruct(), conti_vars)
  nclass_factor <- unlist(data()[, lapply(.SD, function(x){length(unique(x)[!is.na(unique(x))])}),
    .SDcols = factor_vars])
  class01_factor <- unlist(data()[, lapply(.SD, function(x){identical(levels(x), c("0", "1"))}),
    .SDcols = factor_vars])

  validate(
    need(!is.null(class01_factor), "No categorical variables coded as 0, 1 in data")
  )
  factor_01vars <- factor_vars[class01_factor]
  factor_01_list <- mklst(data_varStruct(), factor_01vars)
  group_vars <- factor_vars[nclass_factor >=2 & nclass_factor <=10 & nclass_factor < nrow(data())]
  group_list <- mklst(data_varStruct(), group_vars)
  except_vars <- factor_vars[nclass_factor >10 | nclass_factor ==1 | nclass_factor == nrow(data())]

  ## non-normal: shapiro test
  f <- function(x) {
    if (diff(range(x, na.rm = T)) == 0) return(F) else return(shapiro.test(x)$p.value <= 0.05)
  }

  non_normal <- ifelse(nrow(data()) <=3 | nrow(data()) >= 5000,
    rep(F, length(conti_vars)),
    sapply(conti_vars, function(x){f(data()[[x]])})
  )
  return(list(factor_vars = factor_vars, factor_list = factor_list, conti_vars = conti_vars,
    conti_list = conti_list, factor_01vars = factor_01vars,
    factor_01_list = factor_01_list, group_list = group_list,
    except_vars = except_vars, non_normal = non_normal)
  )
})

out.tb1 <- callModule(tb1simple2, "tb1", data = data, matdata = matdata, data_label = data.label,
  data_varStruct = NULL, vlist = vlist,
  group_var = reactive(mat.info())$group_var)

output$table1_original <- renderDT({
  tb <- out.tb1()$original$table
  cap <- out.tb1()$original$caption
  out <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
  return(out)
})

output$table1_ps <- renderDT({
  tb <- out.tb1()$ps$table
  cap <- out.tb1()$ps$caption
  out <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
  return(out)
})

output$table1_iptw <- renderDT({
  tb <- out.tb1()$iptw$table
  cap <- out.tb1()$iptw$caption
  out <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)

```

```

    return(out)
  })
}

```

---

 tb1simple2

*tb1simple2: tb1 module for propensity score analysis for reactive data*


---

## Description

tb1 module for propensity score analysis for reactive data

## Usage

```

tb1simple2(
  input,
  output,
  session,
  data,
  matdata,
  data_label,
  data_varStruct = NULL,
  vlist,
  group_var,
  showAllLevels = T
)

```

## Arguments

input	input
output	output
session	session
data	Original reactive data with propensity score
matdata	Matching reactive data
data_label	Reactive data label
data_varStruct	List of variable structure, Default: NULL
vlist	List including factor/continuous/binary/except/non-normal variables
group_var	Group variable to run propensity score analysis.
showAllLevels	Show All label information with 2 categorical variables, Default: T

## Details

Table 1 module server for propensity score analysis

## Value

Table 1 with original data/matching data/IPTW data

**See Also**

[CreateTableOneJS svydesign](#)

**Examples**

```

library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
library(haven);library(survey)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      FilePsInput("datafile"),
      tb1simpleUI("tb1")
    ),
    mainPanel(
      DTOutput("table1_original"),
      DTOutput("table1_ps"),
      DTOutput("table1_iptw")
    )
  )
)

server <- function(input, output, session) {

  mat.info <- callModule(FilePs, "datafile")

  data <- reactive(mat.info())$data
  matdata <- reactive(mat.info())$matdata
  data.label <- reactive(mat.info())$data.label

  vlist <- eventReactive(mat.info(), {
    mklist <- function(varlist, vars){
      lapply(varlist,
        function(x){
          inter <- intersect(x, vars)
          if (length(inter) == 1){
            inter <- c(inter, "")
          }
          return(inter)
        })
    }
  })
  factor_vars <- names(data())[data()[, lapply(.SD, class) %in% c("factor", "character")]]
  factor_list <- mklist(data_varStruct(), factor_vars)
  conti_vars <- setdiff(names(data()), c(factor_vars, "pscore", "iptw"))
  conti_list <- mklist(data_varStruct(), conti_vars)
  nclass_factor <- unlist(data()[, lapply(.SD, function(x){length(unique(x)[!is.na(unique(x)]))}),
    .SDcols = factor_vars])
  class01_factor <- unlist(data()[, lapply(.SD, function(x){identical(levels(x), c("0", "1"))}),
    .SDcols = factor_vars])
  validate(
    need(!is.null(class01_factor), "No categorical variables coded as 0, 1 in data")
  )
}

```

```

    factor_01vars <- factor_vars[class01_factor]
    factor_01_list <- mklst(data_varStruct(), factor_01vars)
    group_vars <- factor_vars[nclass_factor >=2 & nclass_factor <=10 & nclass_factor < nrow(data())]
    group_list <- mklst(data_varStruct(), group_vars)
    except_vars <- factor_vars[nclass_factor>10 | nclass_factor==1 | nclass_factor==nrow(data())]

    ## non-normal: shapiro test
    f <- function(x) {
      if (diff(range(x, na.rm = T)) == 0) return(F) else return(shapiro.test(x)$p.value <= 0.05)
    }

    non_normal <- ifelse(nrow(data()) <=3 | nrow(data()) >= 5000,
                        rep(F, length(conti_vars)),
                        sapply(conti_vars, function(x){f(data()[[x]])})
    )
    return(list(factor_vars = factor_vars, factor_list = factor_list, conti_vars = conti_vars,
              conti_list = conti_list, factor_01vars = factor_01vars,
              factor_01_list = factor_01_list, group_list = group_list,
              except_vars = except_vars, non_normal = non_normal)
    )
  })
})

out.tb1 <- callModule(tb1simple2, "tb1", data = data, matdata = matdata, data_label = data.label,
                    data_varStruct = NULL, vlist = vlist,
                    group_var = reactive(mat.info())$group_var)

output$table1_original <- renderDT({
  tb <- out.tb1()$original$table
  cap <- out.tb1()$original$caption
  out <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
  return(out)
})

output$table1_ps <- renderDT({
  tb <- out.tb1()$ps$table
  cap <- out.tb1()$ps$caption
  out <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
  return(out)
})

output$table1_iptw <- renderDT({
  tb <- out.tb1()$iptw$table
  cap <- out.tb1()$iptw$caption
  out <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
  return(out)
})
}

```

**Description**

Table 1 module UI for propensity score analysis.

**Usage**

```
tb1simpleUI(id)
```

**Arguments**

```
id          id
```

**Details**

tb1 module UI for propensity score analysis

**Value**

Table 1 UI for propensity score analysis

**Examples**

```
library(shiny);library(DT);library(data.table);library(readxl);library(jstable)
library(haven);library(survey)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      FilePsInput("datafile"),
      tb1simpleUI("tb1")
    ),
    mainPanel(
      DTOutput("table1_original"),
      DTOutput("table1_ps"),
      DTOutput("table1_iptw")
    )
  )
)

server <- function(input, output, session) {

  mat.info <- callModule(FilePs, "datafile")

  data <- reactive(mat.info())$data
  matdata <- reactive(mat.info())$matdata
  data.label <- reactive(mat.info())$data.label

  vlist <- eventReactive(mat.info(), {
    mklist <- function(varlist, vars){
      lapply(varlist,
        function(x){
          inter <- intersect(x, vars)
          if (length(inter) == 1){
```



```

        inter <- c(inter, "")
      }
      return(inter)
    })
  }
  factor_vars <- names(data())[data()[, lapply(.SD, class) %in% c("factor", "character")]]
  factor_list <- mklist(data_varStruct(), factor_vars)
  conti_vars <- setdiff(names(data()), c(factor_vars, "pscore", "iptw"))
  conti_list <- mklist(data_varStruct(), conti_vars)
  nclass_factor <- unlist(data()[, lapply(.SD, function(x){length(unique(x)[!is.na(unique(x))])}),
    .SDcols = factor_vars])
  class01_factor <- unlist(data()[, lapply(.SD, function(x){identical(levels(x), c("0", "1"))}),
    .SDcols = factor_vars])
  validate(
    need(!is.null(class01_factor), "No categorical variables coded as 0, 1 in data")
  )
  factor_01vars <- factor_vars[class01_factor]
  factor_01_list <- mklist(data_varStruct(), factor_01vars)
  group_vars <- factor_vars[nclass_factor >=2 & nclass_factor <=10 & nclass_factor < nrow(data())]
  group_list <- mklist(data_varStruct(), group_vars)
  except_vars <- factor_vars[nclass_factor>10 | nclass_factor==1 | nclass_factor==nrow(data())]

  ## non-normal: shapiro test
  f <- function(x) {
    if (diff(range(x, na.rm = T)) == 0) return(F) else return(shapiro.test(x)$p.value <= 0.05)
  }

  non_normal <- ifelse(nrow(data()) <=3 | nrow(data()) >= 5000,
    rep(F, length(conti_vars)),
    sapply(conti_vars, function(x){f(data()[[x]])})
  )
  return(list(factor_vars = factor_vars, factor_list = factor_list,
    conti_vars = conti_vars, conti_list = conti_list, factor_01vars = factor_01vars,
    factor_01_list = factor_01_list, group_list = group_list,
    except_vars = except_vars, non_normal = non_normal)
  )
})

out.tb1 <- callModule(tb1simple2, "tb1", data = data, matdata = matdata, data_label = data.label,
  data_varStruct = NULL, vlist = vlist,
  group_var = reactive(mat.info())$group_var)

output$table1_original <- renderDT({
  tb <- out.tb1()$original$table
  cap <- out.tb1()$original$caption
  out <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
  return(out)
})

output$table1_ps <- renderDT({
  tb <- out.tb1()$ps$table
  cap <- out.tb1()$ps$caption

```

```

    out <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
    return(out)
  })

  output$table1_iptw <- renderDT({
    tb <- out.tb1()$iptw$table
    cap <- out.tb1()$iptw$caption
    out <- datatable(tb, rownames = T, extension= "Buttons", caption = cap)
    return(out)
  })
}

```

---

timeROChelper

*timeROChelper: Helper function for timerocModule*


---

## Description

Helper function for timerocModule

## Usage

```

timeROChelper(
  var.event,
  var.time,
  vars.ind,
  t,
  data,
  design.survey = NULL,
  id.cluster = NULL
)

```

## Arguments

var.event	event
var.time	time
vars.ind	independent variable
t	time
data	data
design.survey	survey data, Default: NULL
id.cluster	cluster variable if marginal model, Default: NULL

## Details

Helper function for timerocModule

**Value**

timeROC and coxph object

**See Also**

[coxph](#) [svycoxph](#) [predict](#) [timeROC](#)

**Examples**

```
#library(survival)
#timeROChelper("status", "time", c("age", "sex"), t = 365, data = lung)
```

---

timerocModule

*timerocModule: shiny module server for time-dependent roc analysis*

---

**Description**

shiny module server for time-dependent roc analysis

shiny module server for time-dependent roc analysis- input number of model as integer

**Usage**

```
timerocModule(  
  input,  
  output,  
  session,  
  data,  
  data_label,  
  data_varStruct = NULL,  
  nfactor.limit = 10,  
  design.survey = NULL,  
  id.cluster = NULL,  
  iid = T,  
  NRIIDI = T  
)
```

```
timerocModule2(  
  input,  
  output,  
  session,  
  data,  
  data_label,  
  data_varStruct = NULL,  
  nfactor.limit = 10,  
  design.survey = NULL,  
  id.cluster = NULL,  
  iid = T,
```

```

  NRIIDI = T
)

```

### Arguments

input	input
output	output
session	session
data	Reactive data
data_label	Reactive data label
data_varStruct	Reactive List of variable structure, Default: NULL
nfactor.limit	nlevels limit in factor variable, Default: 10
design.survey	Reactive survey data. default: NULL
id.cluster	Reactive cluster variable if marginal model, Default: NULL
iid	logical, get CI of AUC, Default: T
NRIIDI	logical, get NRI & IDI, Default: T

### Details

shiny module server for time-dependent roc analysis

shiny module server for time dependent roc analysis- input number of model as integer

### Value

shiny module server for time-dependent roc analysis

shiny module server for time dependent roc analysis- input number of model as integer

### See Also

[quantile setkey data.table rbindlist](#)

[quantile setkey data.table rbindlist](#)

### Examples

```

library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
library(timeROC);library(survIDINRI)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      timerocUI("timeroc")
    ),
    mainPanel(
      plotOutput("plot_timeroc"),
      ggplotdownUI("timeroc"),
      DTOutput("table_timeroc")
    )
  )
)

```

```

)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- jstable::mk.lev(mtcars)

  out_timeroc <- callModule(timerocModule, "timeroc", data = data, data_label = data.label,
                           data_varStruct = NULL)

  output$plot_timeroc <- renderPlot({
    print(out_timeroc())$plot
  })

  output$table_timeroc <- renderDT({
    datatable(out_timeroc())$tb, rownames=F, editable = F, extensions= "Buttons",
              caption = "ROC results",
              options = c(jstable::opt.tbreg("roctable"), list(scrollX = TRUE)))
  })
}
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
library(timeroc);library(survIDINRI)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      timerocUI("timeroc")
    ),
    mainPanel(
      plotOutput("plot_timeroc"),
      ggplotdownUI("timeroc"),
      DTOutput("table_timeroc")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- jstable::mk.lev(mtcars)

  out_timeroc <- callModule(timerocModule2, "timeroc", data = data, data_label = data.label,
                           data_varStruct = NULL)

  output$plot_timeroc <- renderPlot({
    print(out_timeroc())$plot
  })

  output$table_timeroc <- renderDT({
    datatable(out_timeroc())$tb, rownames=F, editable = F, extensions= "Buttons",
              caption = "ROC results",
              options = c(jstable::opt.tbreg("roctable"), list(scrollX = TRUE)))
  })
}

```

---

`timerocUI`*timerocUI: shiny module UI for time-dependent roc analysis*

---

**Description**

Shiny module UI for time-dependent roc analysis

**Usage**

```
timerocUI(id)
```

**Arguments**

```
id          id
```

**Details**

Shiny module UI for time-dependent roc analysis

**Value**

Shiny module UI for time-dependent roc analysis

**Examples**

```
library(shiny);library(DT);library(data.table);library(jstable);library(ggplot2)
library(timeROC);library(survIDINRI)
ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      timerocUI("timeroc")
    ),
    mainPanel(
      plotOutput("plot_timeroc"),
      ggplotdownUI("timeroc"),
      DTOutput("table_timeroc")
    )
  )
)

server <- function(input, output, session) {

  data <- reactive(mtcars)
  data.label <- jstable::mk.lev(mtcars)

  out_timeroc <- callModule(timerocModule, "timeroc", data = data, data_label = data.label,
                           data_varStruct = NULL)

  output$plot_timeroc <- renderPlot({
    print(out_timeroc())$plot
  })
}
```

```

    })
    output$table_timeroc <- renderDT({
      datatable(out_timeroc()$tb, rownames=F, editable = F, extensions= "Buttons",
        caption = "ROC results",
        options = c(jstable::opt.tbreg("roctable"), list(scrollX = TRUE)))
    })
  }
}

```

---

timeROC_table	<i>timeROC_table: extract AUC information from list of timeROChelper object.</i>
---------------	--

---

### Description

extract AUC information from list of timeROChelper object.

### Usage

```
timeROC_table(ListModel, dec.auc = 3, dec.p = 3)
```

### Arguments

ListModel	list of timeROChelper object
dec.auc	digits for AUC, Default: 3
dec.p	digits for p value, Default: 3

### Details

extract AUC information from list of timeROChelper object.

### Value

table of AUC information

### See Also

[confint.data.table](#)

### Examples

```

#library(survival)
#list.timeROC <- lapply(list("age", c("age", "sex")),
#
#       function(x){
#         timeROChelper("status", "time", x, t = 365, data = lung)
#       })
#timeROC_table(list.timeROC)

```

# Index

ci.auc, [50](#)  
colon, [30](#)  
confint, [71](#)  
cox2.display, [29](#)  
coxModule, [3](#)  
coxph, [29](#), [54](#), [67](#)  
coxUI, [5](#)  
CreateTableOneJS, [59](#), [62](#)  
csvFile, [5](#)  
csvFileInput, [6](#)

data.table, [28](#), [29](#), [50](#), [54](#), [68](#), [71](#)  
dropdownButton, [40](#)

FilePs, [7](#)  
FilePsInput, [9](#)  
FileRepeated, [10](#)  
FileRepeatedInput, [11](#)  
FileSurvey, [12](#)  
FileSurveyInput, [13](#)  
fwrite, [26](#), [28](#), [30](#), [32](#)

geeglm, [46](#), [48](#)  
GEEModuleLinear, [14](#)  
GEEModuleLogistic, [16](#)  
GEEModuleUI, [17](#)  
ggpairsModule, [19](#)  
ggpairsModule2, [20](#)  
ggpairsModuleUI1, [21](#)  
ggpairsModuleUI2, [23](#)  
ggplotdownUI, [24](#)  
ggroc, [46](#), [48](#)  
ggsave, [29](#)

IDI.INF, [54](#)  
IDI.INF.OUT, [54](#)

jsBasicAddin, [25](#)  
jsBasicExtAddin, [25](#)  
jsBasicGadget, [26](#)  
jskm, [29](#)  
jsPropensityAddin, [27](#)  
jsPropensityExtAddin, [27](#)  
jsPropensityGadget, [28](#)  
jsRepeatedAddin, [29](#)  
jsRepeatedExtAddin, [30](#)  
jsRepeatedGadget, [30](#)  
jsSurveyAddin, [31](#)  
jsSurveyExtAddin, [32](#)  
jsSurveyGadget, [32](#)

kaplanModule, [33](#)  
kaplanUI, [35](#)

logistic.display2, [36](#)  
logisticModule2, [37](#)  
lung, [26](#)

match.data, [29](#)  
matchit, [29](#)  
mklist, [38](#)  
mksetdiff, [39](#)  
model.matrix, [54](#)

opt.tb1, [32](#)  
opt.tbreg, [26](#), [28](#), [30](#), [32](#)  
optionUI, [39](#)

pbcr, [28](#)  
predict, [67](#)

quantile, [46](#), [48](#), [68](#)

rbindlist, [50](#), [68](#)  
rcorrp.cens, [41](#)  
reclassificationJS, [40](#)  
regress.display2, [42](#)  
regressModule2, [43](#)  
regressModuleUI, [44](#)  
roc.test, [50](#)  
ROC\_table, [50](#)  
rocModule, [45](#)



rocModule2, 47  
rocUI, 49

scatterServer, 51  
scatterUI, 52  
setkey, 46, 48, 68  
Surv, 29, 54  
survfit, 29  
survIDINRI\_helper, 53  
svycox.display, 29  
svycoxph, 67  
svydesign, 28, 59, 62  
svyglm, 46, 48  
svyjskm, 29  
svykm, 29

tb1module, 54  
tb1module2, 55  
tb1moduleUI, 57  
tb1simple, 58  
tb1simple2, 61  
tb1simpleUI, 63  
theme\_modern, 46, 48  
timeROC, 67  
timeROC\_table, 71  
timeROChelper, 66  
timerocModule, 67  
timerocModule2 (timerocModule), 67  
timerocUI, 70  
tooltipOptions, 40

var\_label, 59