4 ExpoStat module

4.1 Application domain

Data statistics is designed for the statistical description of data. Before selecting the statistical algorithm to build the model, it is necessary to understand the distribution of the data and the socio-demographic characteristics of the population. The results of the statistical description provide the basis for selecting the statistical algorithm to build the model and the potential confounding to be adjusted during the modeling process.

The ExpoStat module provides functions for creating the basic descriptive statistics table, testing normality distribution of variable, screening extreme values, comparing the size between/among groups, and calculating correlation coefficient between variables. The visualization of the statistical results are also provided. It provides six main functions for users : StatTable1, StatNorm, StatExtre, StatDesc, StatComp, and StatCorr.

- StatTable1: Create a statistical description table of demographic characteristics for three common types of epidemiological studies(i.e., cohort, case control, and cross sectional).
- StatNorm: Normality test for continuous variables, and visualizations of whether variables conform to normal distribution are provided.
- StatExtre: Calculating The extreme value of the variable and output the result. The visualization of the result shows the frequency of the extreme value in each subject.
- StatDesc: Calculate the mean, standard deviation, median and quartile of continuous variables, and the count and frequency of discrete variables..
- StatComp: Performing inter-group comparison for continuous variables and visualize whether there is a statistical difference between the different groups.
- StatCorr: Calculate the correlation coefficient between each variable, output the correlation coefficient and P value, and visualize the results.

4.2 Theory

The different “ExpoStat” module is based on different statistics test. Users can easily get the modeling results and their visualization plots with high quality by following the detailed instructions in each step. Only typical statistical methods are included in our module for convenience. For example, the statistical method of normality test is Shapiro-Wilk test, the statistical method of inter-group comparison is Wilcoxon rank test, and the statistical method of calculating the correlation coefficient between each variable is Pearson correlation coefficient or Spearman rank correlation coefficient.

4.3 Work pipeline

Initialize package

Make sure that the required packages is already installed.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exstat", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exstat)
# library(extidy)
library(tidyverse)
```

```r
# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

At first, you need to initialize the calculation environment using a series of initialization functions, e.g., InitCros, InitMo, InitTidy, InitViz, InitBiolink, etc. Here, we use the package "exstat" for data description for example. The detailed information about the functions and returned value will be introcuced in the following chapters.

```r
res <- InitStat()
res
```

```
## <eSet>
##   Public:
##     AddCommand: function (x)
##     AddLog: function (x)
##     clone: function (deep = FALSE)
##     EpiDesign: NULL
##     ExcecutionLog: Complete initializing the ExpoStat module.2022.12.14 14. ...
##     Expo: list
##     FileDirIn: NULL
##     FileDirOut: /home/ubuntu/@changxin/R_Exposome_1.0/output_144651GJAFZA
##     PID: 144651GJAFZA
##     RCommandLog: eSet <- InitVisual(PID = Any ID your like, FileDirOut =  ...
##     VarsDel: NULL
```

Here, we can see that the returned value "res" is an R6 object. It contains an unique program ID of res$PID (e.g., "100737GJMWJA"), which is random generated by the system. Users need to use it in the following step for further data process.

Upload data

After initializing the calculation environment, the second step is to upload local data file for ExpoStat Module. `LoadStat` is provided for this. It has four parameters, PID, UseExample, DataPath and VocaPath. PID is Program ID, which must be the same with the PID generated by `InitStat`. UseExample is a character indicates whether uses example data for analyses, available option include "example#1" for using example data1 and "default" for using data uploaded. DataPath refer to the input file directory, e.g. "D:/test/eg_data.xlsx". It should be noted that the slash symbol is `/`, not `\`. For convenience, here we use example data one for the following step.

```r
res1 <- LoadStat(res$PID,
                 UseExample = "example#1")

res1$Expo$Voca %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	SerialNo_Raw	FullName	GroupName	Lod
Y1	Y1	Y_disc	Outcome	NA
Y2	Y2	Y_cont	Outcome	NA
C1	C1	Cov_1	Demography	NA
C2	C2	Cov_2	Demography	NA
C3	C3	Cov_3	Demography	NA
C4	C4	Cov_4	Demography	NA
C5	C5	Cov_5	Demography	NA
C6	C6	Cov_6	Demography	NA
X1	X1	TE_1	Chemical	0.5
X2	X2	TE_2	Chemical	0.5
X3	X3	TE_3	Chemical	0.5
X4	X4	TE_4	Chemical	0.5
X5	X5	TE_5	Chemical	0.5
X6	X6	TE_6	Chemical	0.5
X7	X7	TE_7	Chemical	0.5
X8	X8	TE_8	Chemical	0.5
X9	X9	CH1	Chemical	5.0
X10	X10	CH2	Chemical	5.0
X11	X11	CH3	Chemical	5.0
X12	X12	CH4	Chemical	5.0

```
res1$Expo$Data %>%
  dplyr::select(SampleID:C2) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SampleID	SubjectID	Group	Y1	Y2	C1	C2
Tr1	S1	train	1	-101	26.86773	25.35056
Tr2	S2	train	0	-51	30.91822	23.94432
Tr3	S3	train	0	-37	25.82186	23.04579
Tr4	S4	train	1	-61	37.97640	21.21191
Tr5	S5	train	0	-28	31.64754	19.53762
Tr6	S6	train	0	-8	25.89766	20.77442
Tr7	S7	train	1	-63	32.43715	27.00009
Tr8	S8	train	0	-35	33.69162	22.13620
Tr9	S9	train	0	-14	32.87891	19.84672
Tr10	S10	train	1	-99	28.47306	29.60787
Tr11	S11	train	0	-60	37.55891	25.27530
Tr12	S12	train	0	-32	31.94922	23.28406
Tr13	S13	train	0	-73	26.89380	27.17545
Tr14	S14	train	0	-18	18.92650	26.65927
Tr15	S15	train	0	-48	35.62465	22.14227
Tr16	S16	train	0	-20	29.77533	30.61831
Tr17	S17	train	0	-9	29.91905	23.23492
Tr18	S18	train	1	-98	34.71918	19.72652
Tr19	S19	train	0	-70	34.10611	23.56680
Tr20	S20	train	0	-36	32.96951	24.62261

Tidy data

After initializing the calculation environment, and upload local data file for ExpoStat Module. Users can choose to clean up the data, including deleting missing values, scale the data, changing the type of the data, interpolation of missing values, and so on.

```
res2 <- DelMiss(res$PID)
res2$Expo$Voca
```

```
## # A tibble: 226 x 5
##    SerialNo SerialNo_Raw FullName GroupName    Lod
##    <chr>    <chr>        <chr>    <chr>      <dbl>
##  1 Y1       Y1           Y_disc   Outcome     NA
##  2 Y2       Y2           Y_cont   Outcome     NA
##  3 C1       C1           Cov_1    Demography  NA
##  4 C2       C2           Cov_2    Demography  NA
##  5 C3       C3           Cov_3    Demography  NA
##  6 C4       C4           Cov_4    Demography  NA
##  7 C5       C5           Cov_5    Demography  NA
##  8 C6       C6           Cov_6    Demography  NA
##  9 X2       X2           TE_2     Chemical     0.5
## 10 X3       X3           TE_3     Chemical     0.5
## # ... with 216 more rows
```

```
res3 <- DelNearZeroVar(res$PID)
res3$Expo$Voca
```

```
## # A tibble: 209 x 5
##    SerialNo SerialNo_Raw FullName GroupName    Lod
##    <chr>    <chr>        <chr>    <chr>      <dbl>
##  1 Y1       Y1           Y_disc   Outcome     NA
##  2 Y2       Y2           Y_cont   Outcome     NA
##  3 C1       C1           Cov_1    Demography  NA
##  4 C2       C2           Cov_2    Demography  NA
##  5 C3       C3           Cov_3    Demography  NA
##  6 C4       C4           Cov_4    Demography  NA
##  7 C5       C5           Cov_5    Demography  NA
##  8 C6       C6           Cov_6    Demography  NA
##  9 X7       X7           TE_7     Chemical     0.5
## 10 X8       X8           TE_8     Chemical     0.5
## # ... with 199 more rows
```

```
res4 <- TransClass(res$PID,
                   Group = F,
                   Vars = "X10",
                   LevelTo = 4)
res4$Expo$Voca
```

```
## # A tibble: 209 x 5
##    SerialNo SerialNo_Raw FullName GroupName    Lod
##    <chr>    <chr>        <chr>    <chr>      <dbl>
##  1 Y1       Y1           Y_disc   Outcome     NA
##  2 Y2       Y2           Y_cont   Outcome     NA
##  3 C1       C1           Cov_1    Demography  NA
##  4 C2       C2           Cov_2    Demography  NA
##  5 C3       C3           Cov_3    Demography  NA
##  6 C4       C4           Cov_4    Demography  NA
```

```
##   7 C5       C5            Cov_5    Demography  NA
##   8 C6       C6            Cov_6    Demography  NA
##   9 X7       X7            TE_7     Chemical     0.5
##  10 X8       X8            TE_8     Chemical     0.5
## # ... with 199 more rows
```

StatTable1

StatTable1 provides sociodemographic information tables for three common epidemiological studies, returning the information to the user based on epidemiological design. EpiDesign/Group/VarsY/VarsC/Missing parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatTable1 can only run successfully after successfully running InitStat and LoadStat functions.

EpiDesign/Group/VarsY/VarsC/Missing parameters can be a character, run `?StatTable1` to see more details.

```
res_table1<-StatTable1(res$PID,
                       EpiDesign = "case.control",
                       Group = F,
                       VarsY = "Y1",
                       VarsC = res1$Expo$Voca %>%
                         dplyr::filter(str_detect(SerialNo_Raw, "C")) %>%
                         .$SerialNo_Raw %>%
                         str_c(collapse = ","),
                       Missing = "ifany")
# res_table1

res_table1<-StatTable1(res$PID,
                       EpiDesign = "cohort",
                       Group = F,
                       VarsY = "Y1",
                       VarsC = "all.c",
                       Missing = "always")
# res_table1
```

StatNorm

StatNorm provides function of normality test for continuous variable.For the time being, only the typical Shapiro-Wilk test is provided to test the normality of variables, and other tests will be added gradually in the future. Group/Vars/Method/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatNorm can only run successfully after successfully running InitStat and LoadStat functions.

Group/Vars/Method/Layout/Brightness/Palette parameters can be a character, run `?StatNorm` to see more details.

```
res_norm <- StatNorm(PID = res$PID,
                     Group = T,
                     Vars =  "all",
                     Method = "shapiro.test",
                     Layout = "rose.chart",
                     Brightness = "light",
                     Palette = "default3")
res_norm
```

```
## $normtable
```

```
## $normtable$StatNorm_by_shapiro.test
## # A tibble: 416 x 5
##    Group Vars       P Method       Normal
##    <chr> <fct>  <dbl> <chr>        <chr>
##  1 Train Y1    3      shapiro.test N
##  2 Train Y2    3      shapiro.test N
##  3 Train C1    0.103  shapiro.test Y
##  4 Train C2    0.0326 shapiro.test Y
##  5 Train C3    3      shapiro.test N
##  6 Train C4    3      shapiro.test N
##  7 Train C5    3      shapiro.test N
##  8 Train C6    3      shapiro.test N
##  9 Train X7    3      shapiro.test N
## 10 Train X8    3      shapiro.test N
## # ... with 406 more rows
##
##
## $NormPlot
## NULL
##
## $NormPlot_para
## NULL
```

StatExtre

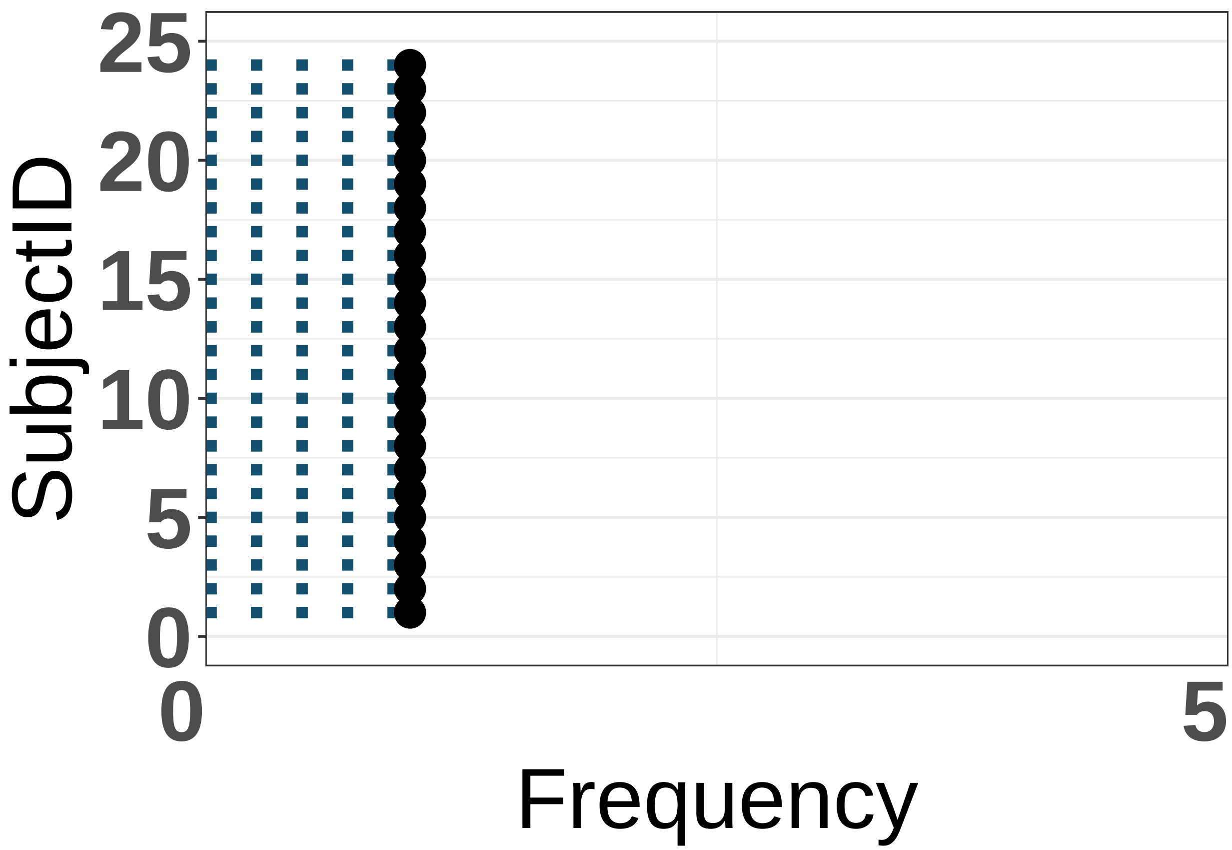
StatExtre provides the functions of calculating The extreme value for the continuous variable. The two parameters 'LimitLow' and 'LimitUpper' can be set to adjust the percentile of the variable. Group/Vars/LimitLow/LimitUpper/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatExtre can only run successfully after successfully running InitStat and LoadStat functions.

Group/Vars/LimitLow/LimitUpper/Layout/Brightness/Palette parameters can be a character, run `?StatExtre` to see more details.

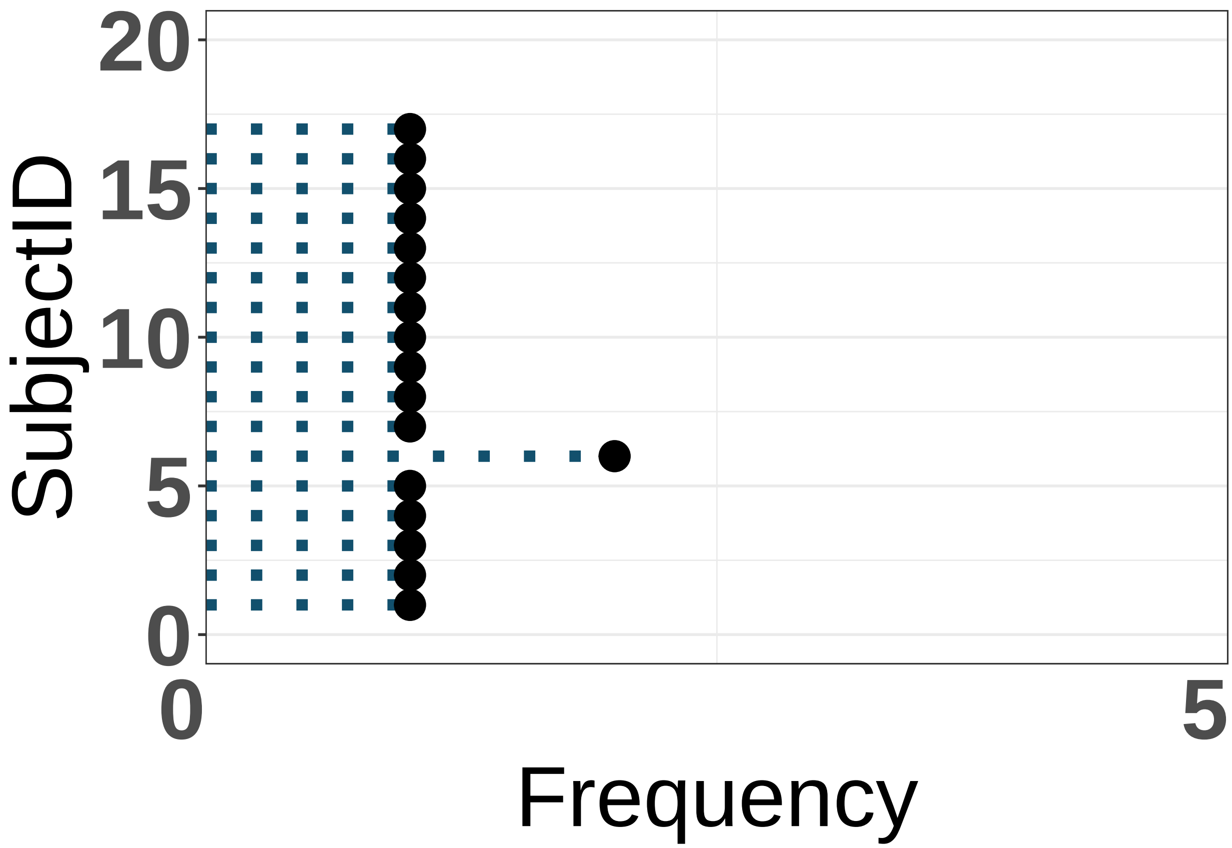
```
res_extre<-StatExtre(PID = res$PID,
                     Group = T,
                     Vars = "X5,X6,X7,X8,X9",
                     LimitLow = 0.025,
                     LimitUpper = 0.975,
                     Layout = "column.points",
                     Brightness = "dark",
                     Palette = "default2")
res_extre
```

```
## $Extretable
## # A tibble: 42 x 5
##    Group SerialNo value SubjectID SampleID
##    <chr> <chr>    <dbl> <chr>     <chr>
##  1 Train X7       -5    S106      Tr106
##  2 Train X7       -4.91 S150      Tr150
##  3 Train X7       -4.86 S102      Tr102
##  4 Train X7       -4.59 S5        Tr5
##  5 Train X7        4.59 S138      Tr138
##  6 Train X7        4.60 S38       Tr38
##  7 Train X7        4.78 S12       Tr12
```

```
##   8 Train X7         5     S1         Tr1
##   9 Train X8        -5     S49        Tr49
## 10 Train X8        -4.94 S116       Tr116
## # ... with 32 more rows
##
## $ExtrePlot
## $ExtrePlot$VizStatExtre_Train_ColumnPoints_dark_default2
```




```
##
## $ExtrePlot$VizStatExtre_Test_ColumnPoints_dark_default2
```




```
##
##
## $ExtrePlot_para
## $ExtrePlot_para$VizStatExtre_Train_ColumnPoints_dark_default2
## $ExtrePlot_para$VizStatExtre_Train_ColumnPoints_dark_default2$width
## [1] 37
##
## $ExtrePlot_para$VizStatExtre_Train_ColumnPoints_dark_default2$height
## [1] 56
##
##
## $ExtrePlot_para$VizStatExtre_Test_ColumnPoints_dark_default2
## $ExtrePlot_para$VizStatExtre_Test_ColumnPoints_dark_default2$width
## [1] 37
##
## $ExtrePlot_para$VizStatExtre_Test_ColumnPoints_dark_default2$height
## [1] 54
```

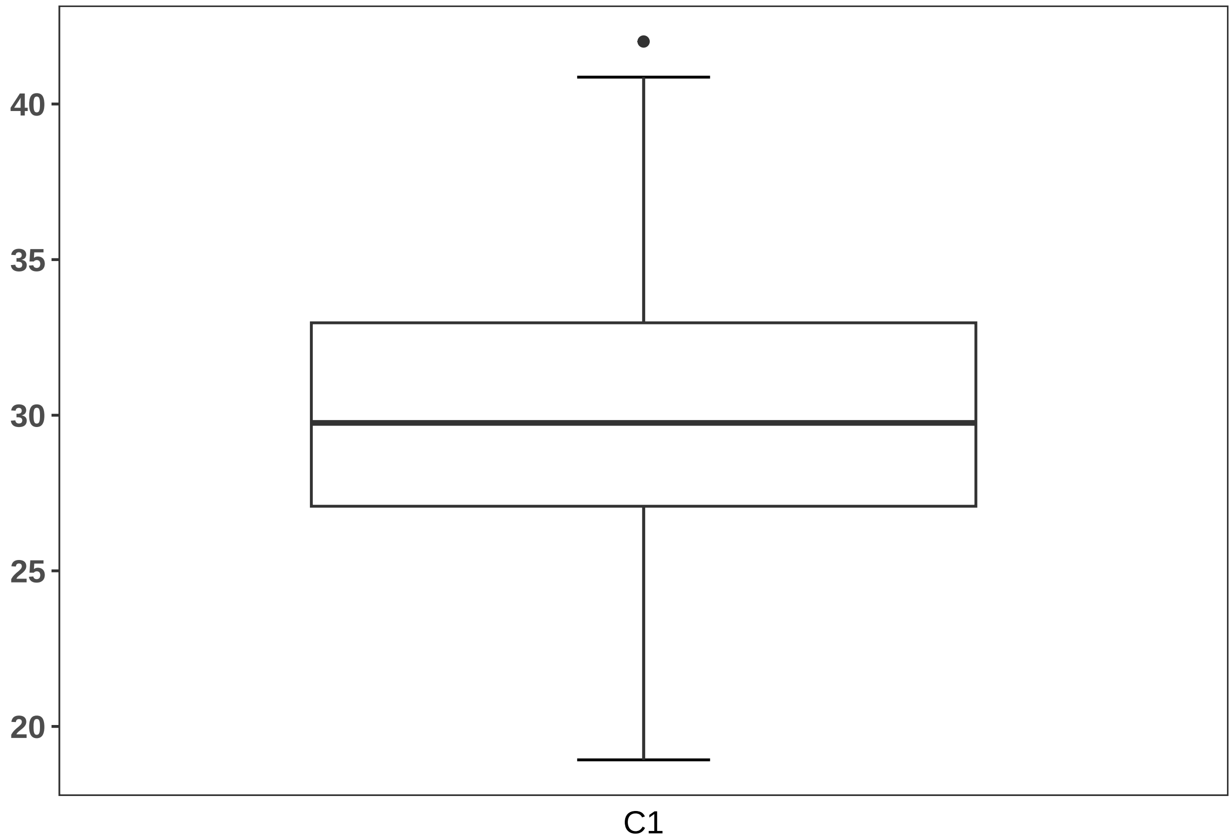
StatDesc

StatDesc provides the description of the size of a variable. Different types of variables have different size descriptions.The module calculates the mean, standard deviation, median, quartile and other information of continuous variables, the count and frequency of discrete variables. The visualization of this part is different from the visualization of the result statistics of other functions. This part only provides the box diagram and violin diagram to represent the median quartile of continuous variables. Group/Vars/VarsBy/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatDesc can only run successfully after successfully running InitStat and LoadStat functions.

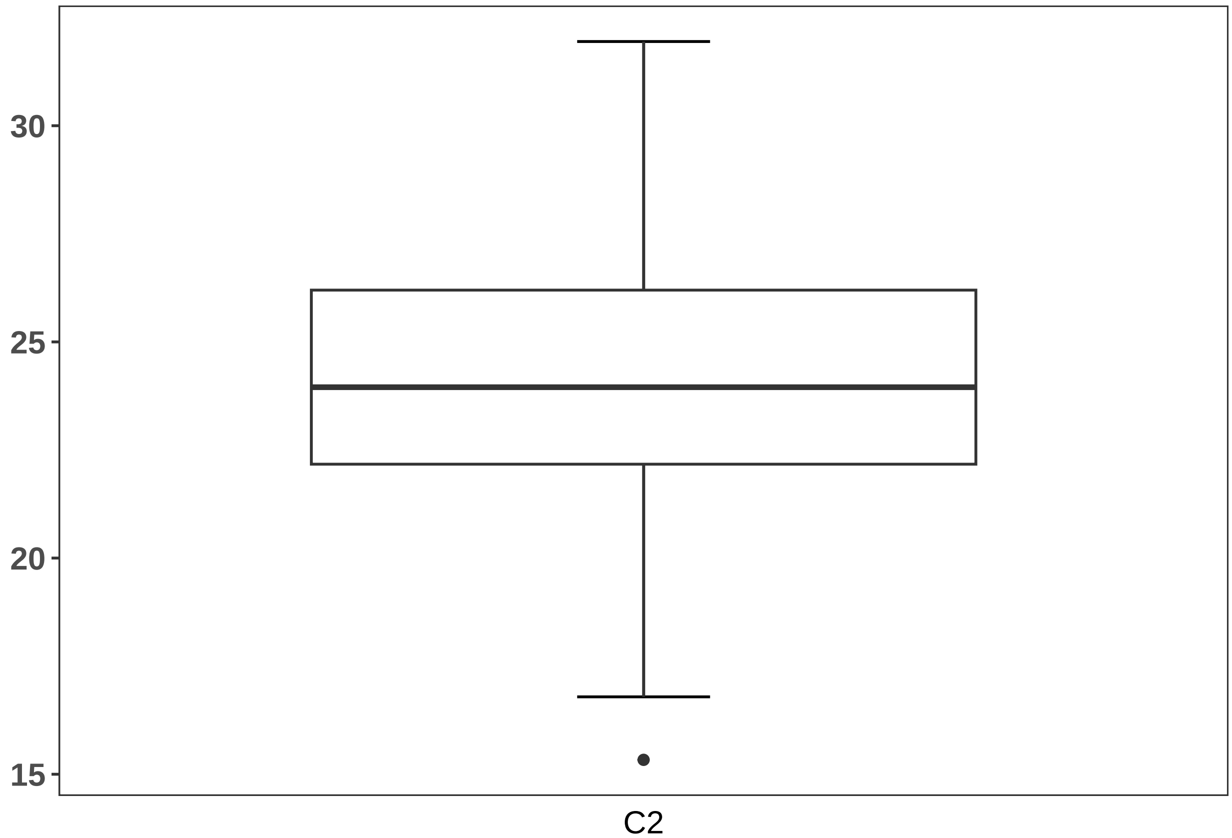
Group/Vars/VarsBy/Layout/Brightness/Palette parameters can be a character, run `?StatDesc` to see more details.

```
res_desc <- StatDesc(PID = res$PID,
                     Group = T,
                     Vars = "C1,C2,X5,X6,X7,X8,X9",
                     VarsBy = NULL,
                     Layout = "box",
                     Brightness = "dark",
                     Palette = "default2")
res_desc
```

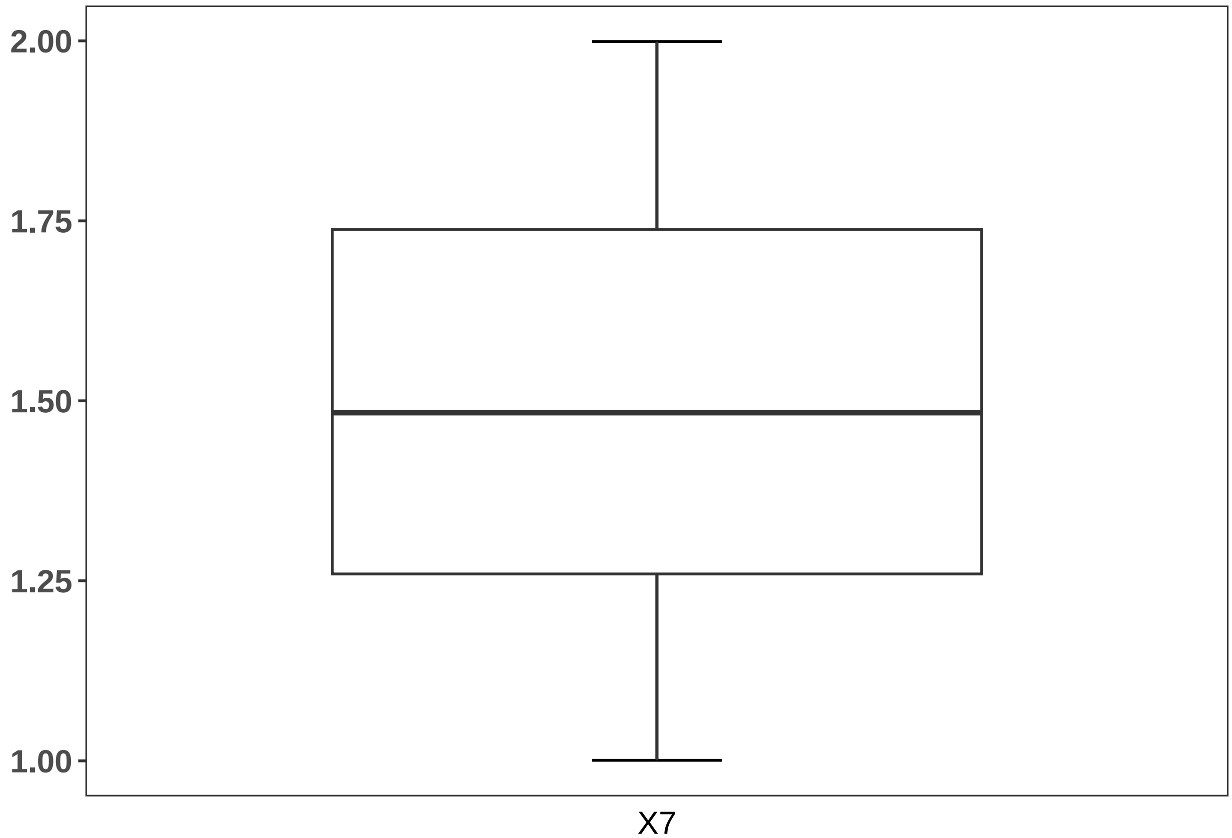
```
## $Desctable
## $Desctable$`/Stat_DescNum_by_`
##    Group variable   n    min    max median     q1     q3   iqr   mad   mean
## 1  Train       C1 150 18.927 42.008 29.753 27.075 32.970 5.895 4.326 30.109
## 2   Test       C1  91 18.927 40.863 29.919 27.145 33.300 6.155 4.479 30.142
## 3  Train       C2 150 15.333 31.948 23.952 22.171 26.198 4.026 2.914 24.136
## 4   Test       C2  91 17.133 31.948 24.006 22.123 25.715 3.592 2.773 24.130
## 5  Train       X7 150  1.001  1.999  1.484  1.260  1.738 0.478 0.367  1.500
## 6   Test       X7  91  1.015  1.953  1.515  1.292  1.702 0.410 0.312  1.504
## 7  Train       X8 150  1.000  1.992  1.460  1.256  1.748 0.492 0.360  1.486
## 8   Test       X8  91  1.018  1.992  1.499  1.305  1.734 0.429 0.309  1.508
## 9  Train       X9 150  4.493 31.199  8.174  6.782 10.386 3.604 2.158 10.095
## 10  Test       X9  91  5.147 24.346  7.973  6.704  9.268 2.564 1.898  9.624
##       sd    se    ci
## 1  4.521 0.369 0.729
## 2  4.765 0.500 0.992
## 3  3.068 0.250 0.495
## 4  2.983 0.313 0.621
## 5  0.289 0.024 0.047
## 6  0.272 0.029 0.057
## 7  0.291 0.024 0.047
## 8  0.265 0.028 0.055
## 9  5.188 0.424 0.837
## 10 4.818 0.505 1.003
##
##
## $DescPlot
## $DescPlot$VizStatDesc_Train_Box_C1
```



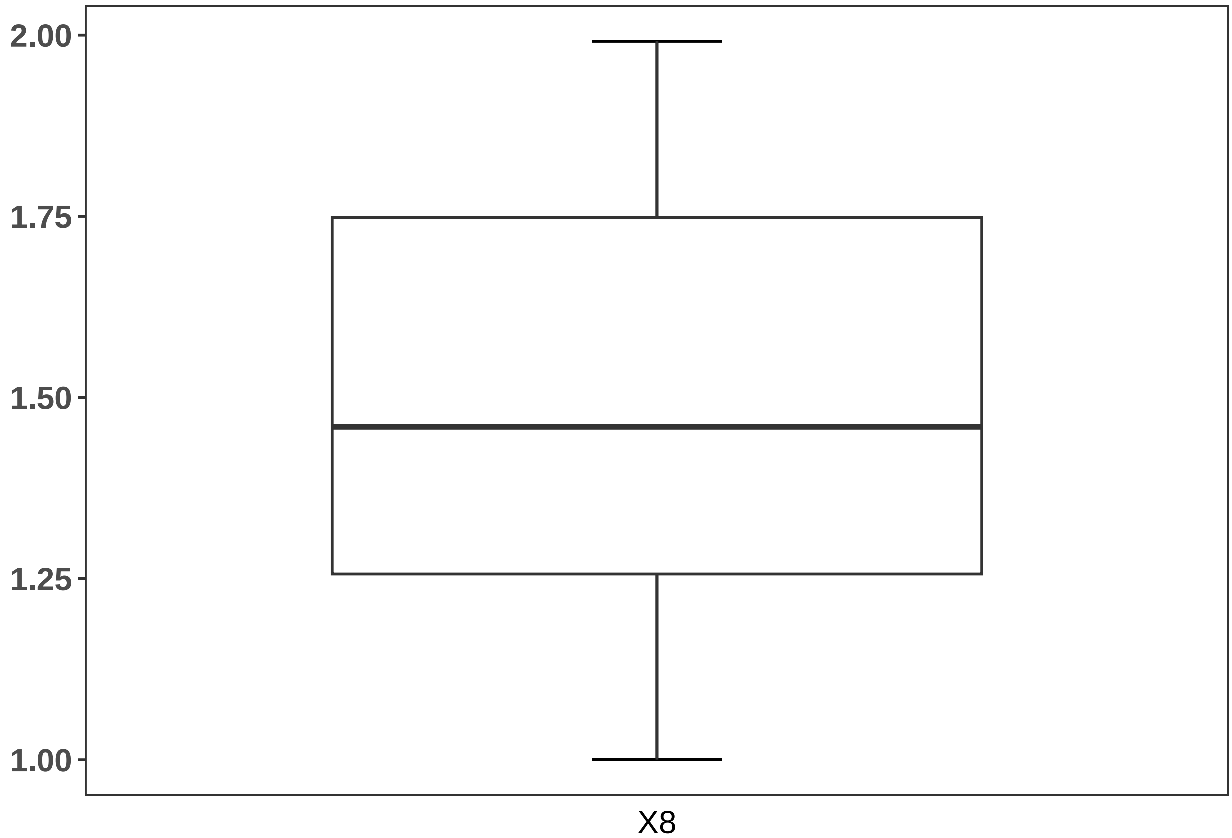

```
##
## $DescPlot$VizStatDesc_Train_Box_C2
```



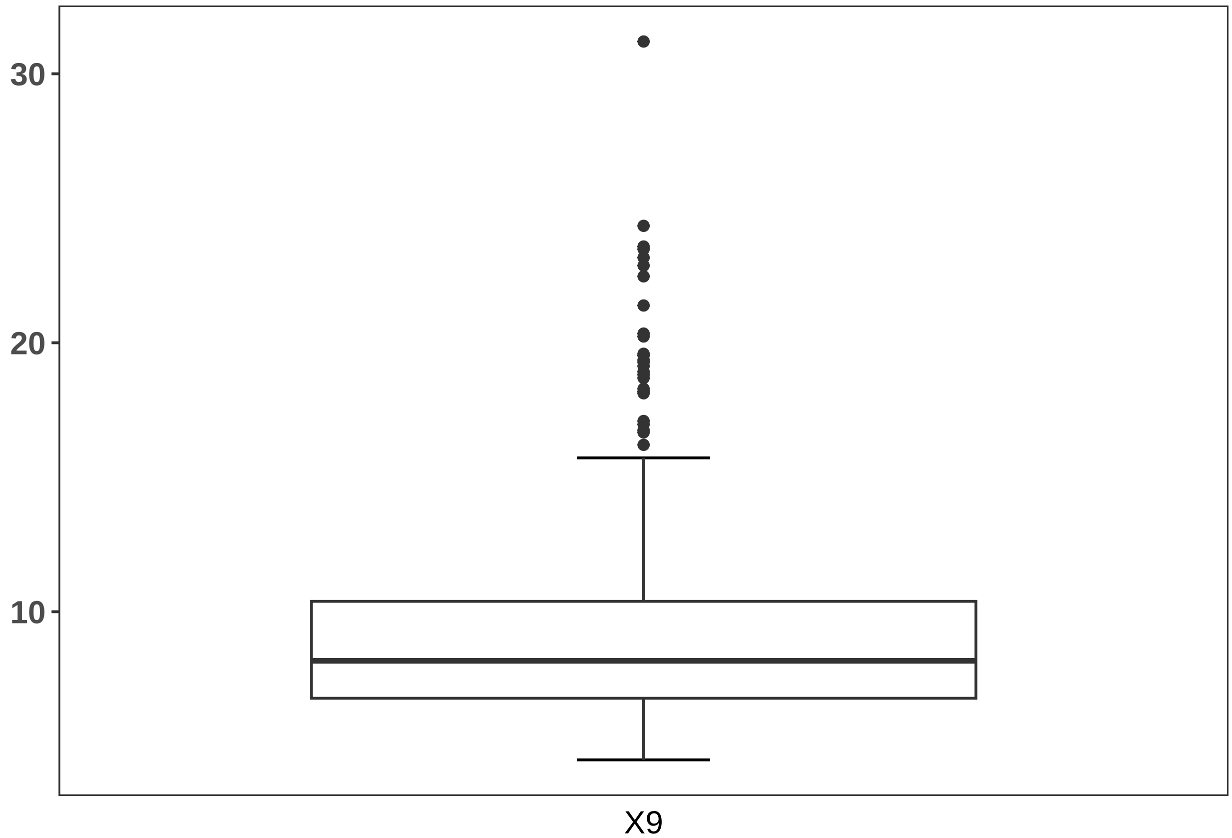

```
##
## $DescPlot$VizStatDesc_Train_Box_X7
```



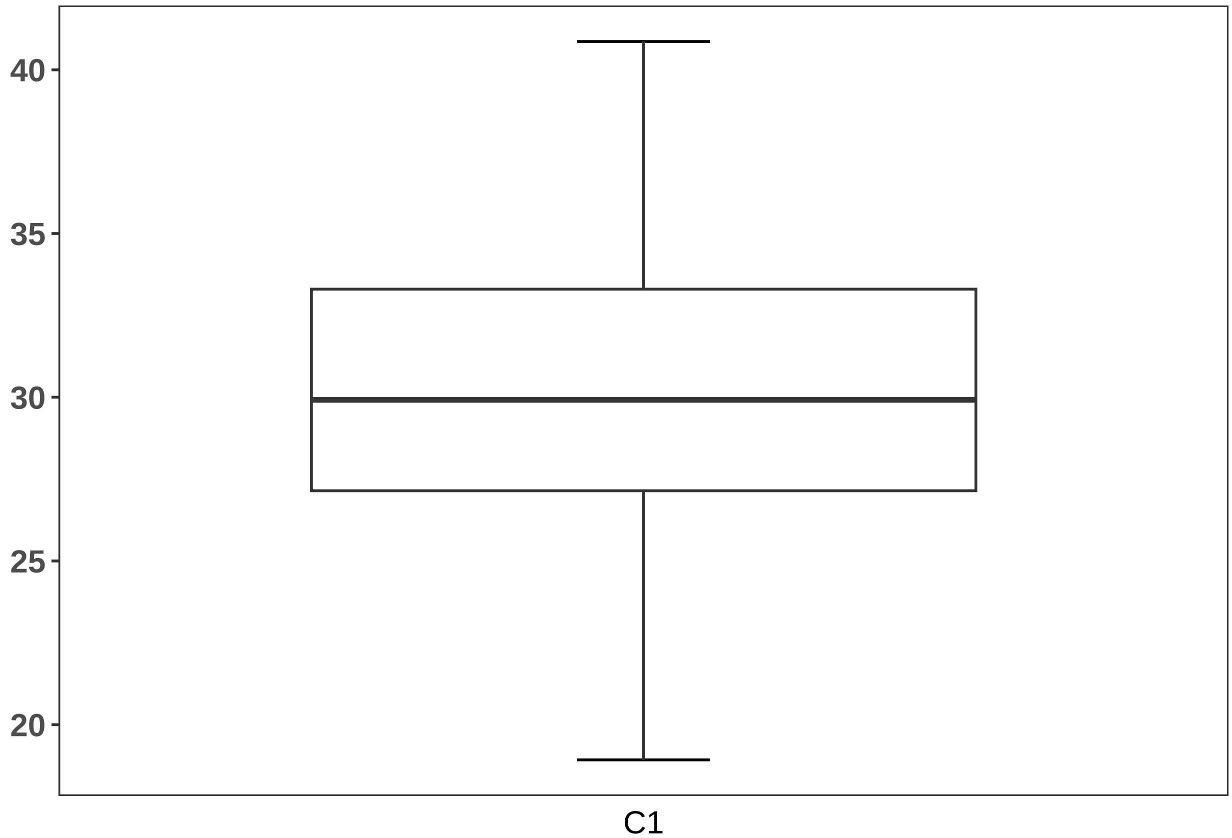

```
##
## $DescPlot$VizStatDesc_Train_Box_X8
```



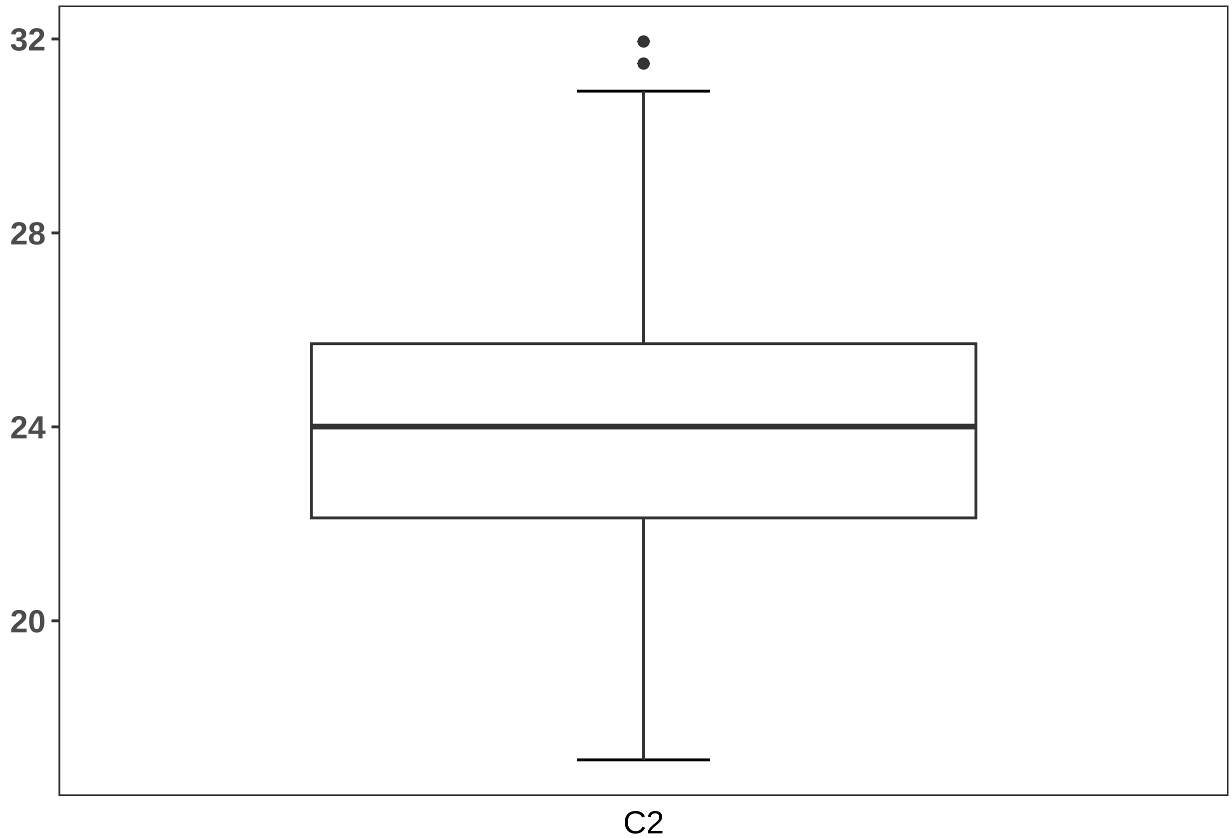

```
##
## $DescPlot$VizStatDesc_Train_Box_X9
```



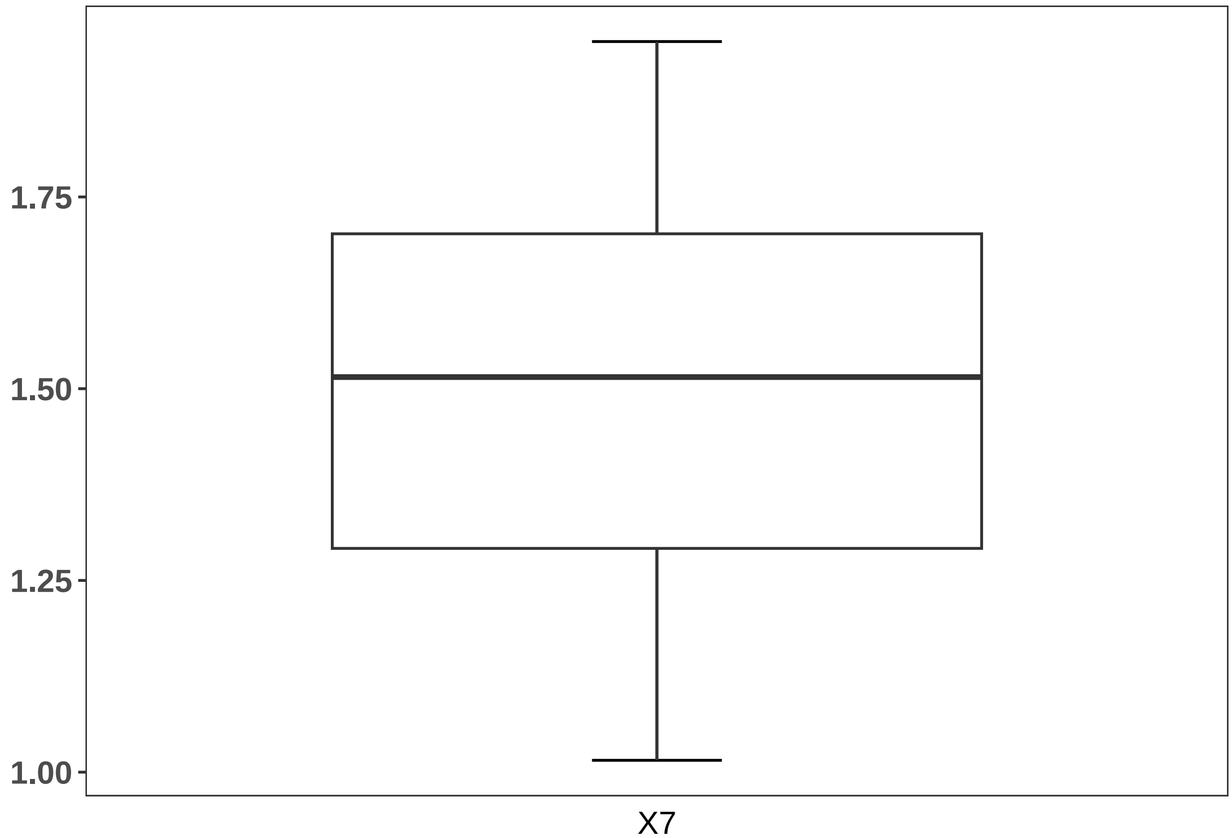

```
##
## $DescPlot$VizStatDesc_Test_Box_C1
```



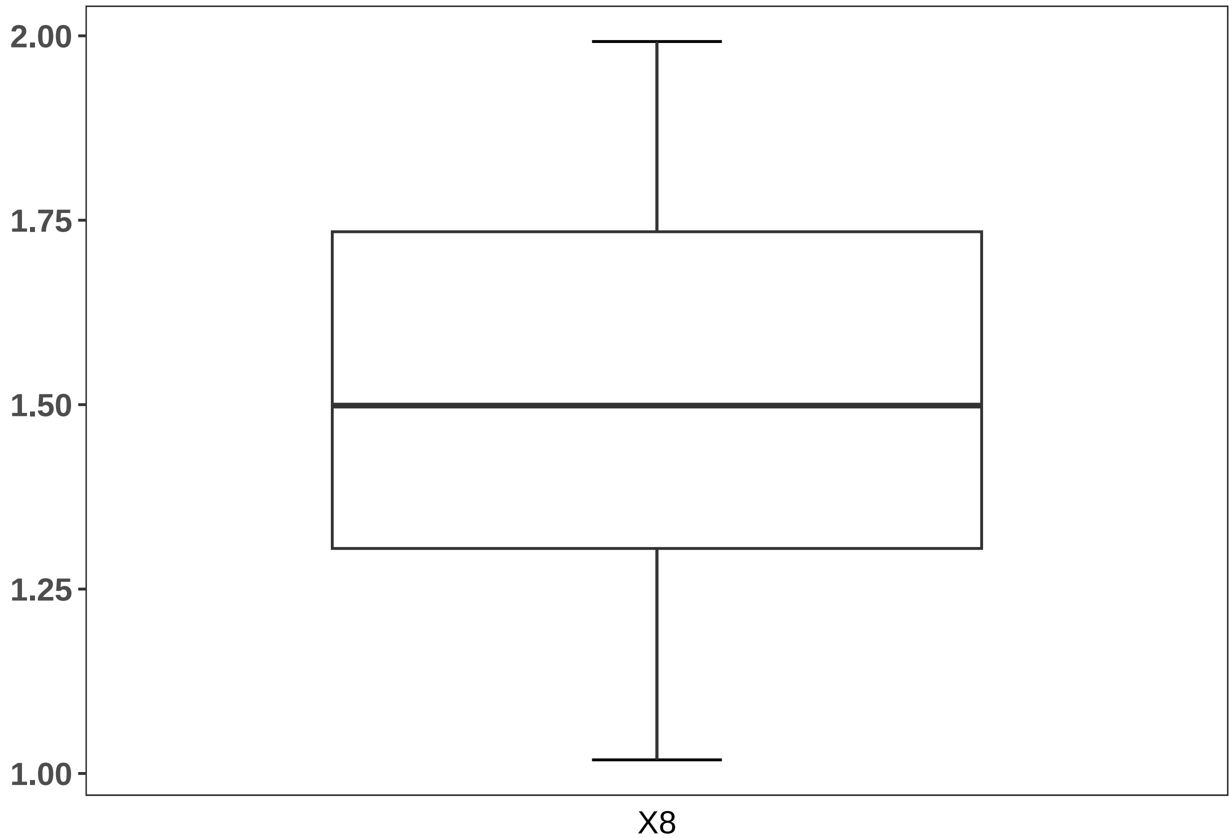

```
##
## $DescPlot$VizStatDesc_Test_Box_C2
```



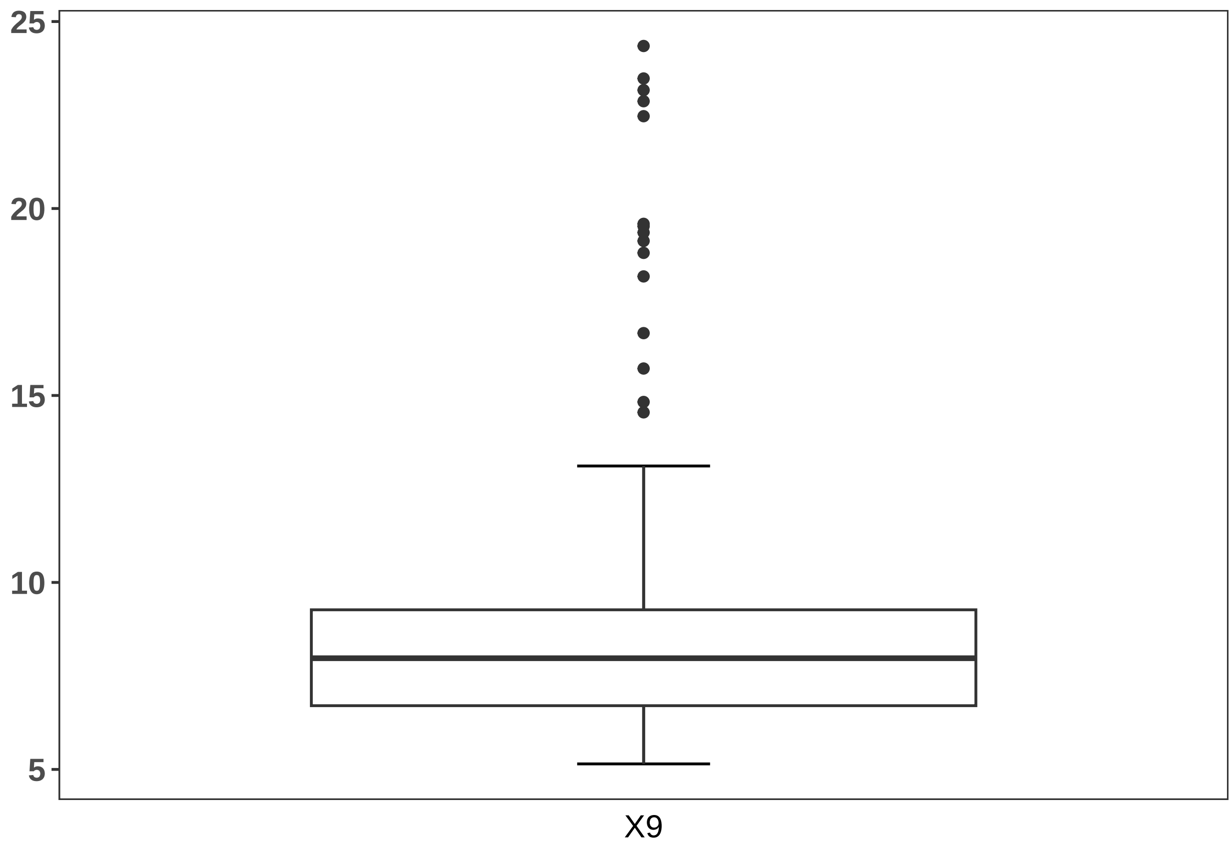

```
##
## $DescPlot$VizStatDesc_Test_Box_X7
```




```
##
## $DescPlot$VizStatDesc_Test_Box_X8
```




```
##
## $DescPlot$VizStatDesc_Test_Box_X9
```




```
##
##
## $DescPlot_para
## $DescPlot_para$VizStatDesc_Train_Box_C1
## $DescPlot_para$VizStatDesc_Train_Box_C1$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_C1$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Train_Box_C2
## $DescPlot_para$VizStatDesc_Train_Box_C2$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_C2$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Train_Box_X7
## $DescPlot_para$VizStatDesc_Train_Box_X7$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_X7$height
## [1] 15
##
```

```
##
## $DescPlot_para$VizStatDesc_Train_Box_X8
## $DescPlot_para$VizStatDesc_Train_Box_X8$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_X8$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Train_Box_X9
## $DescPlot_para$VizStatDesc_Train_Box_X9$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_X9$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_C1
## $DescPlot_para$VizStatDesc_Test_Box_C1$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_C1$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_C2
## $DescPlot_para$VizStatDesc_Test_Box_C2$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_C2$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_X7
## $DescPlot_para$VizStatDesc_Test_Box_X7$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_X7$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_X8
## $DescPlot_para$VizStatDesc_Test_Box_X8$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_X8$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_X9
## $DescPlot_para$VizStatDesc_Test_Box_X9$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_X9$height
```

```
## [1] 15
```

StatComp

StatComp provides the function of size comparison between groups. For the time being, only the typical Wilcoxon rank test is provided to test the difference between groups, and other tests will be added gradually in the future. Group/Task/Vars/VarsBy/Method/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatDesc can only run successfully after successfully running InitStat and LoadStat functions.

Group/Task/Vars/VarsBy/Method/Layout/Brightness/Palette parameters can be a character, run `?StatComp` to see more details.

```
#res_comp <- StatComp(PID = res$PID,
#                      Group = T,
#                      Task = "mean",
#                      Vars = "X5,X6,X7,X8,X9",
#                      VarsBy = "Y1",
#                      Method = "wilcox",
#                      Layout = "density",
#                      Brightness = "dark",
#                      Palette = "default1")
```

StatCorr

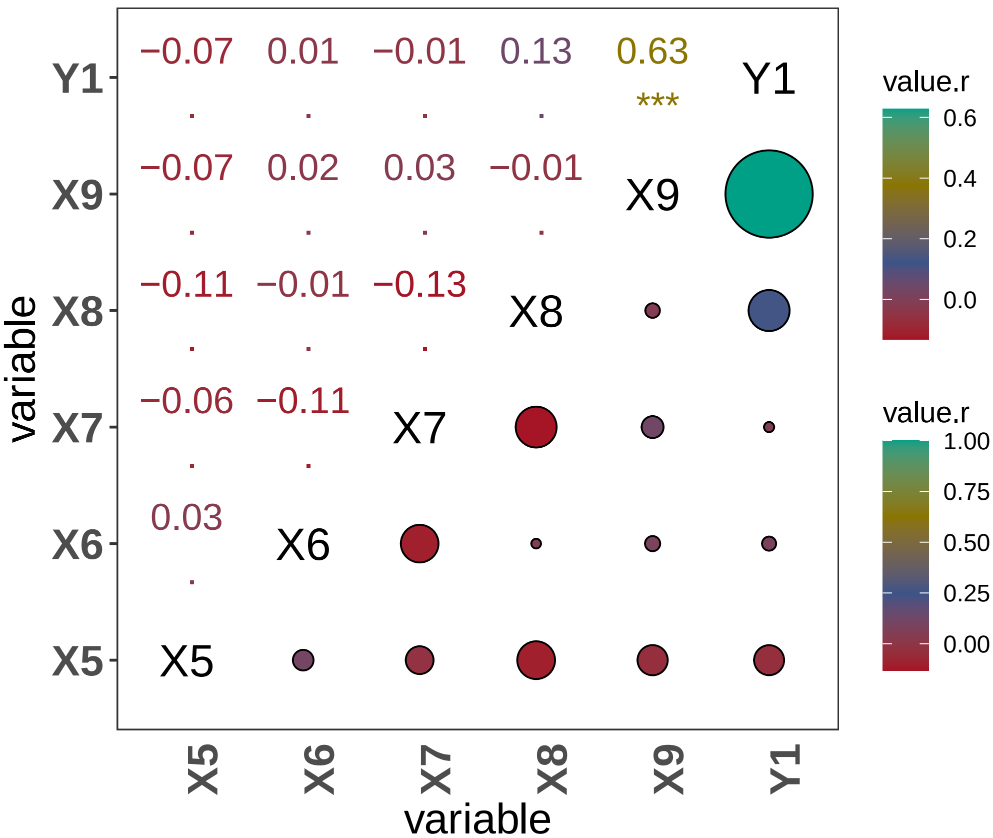
StatCorr provides the function of calculating of correlation coefficient between variables. For the time being, there are two typical statistics method Pearson correlation coefficient or Spearman rank correlation coefficient is provided to calculating of correlation, and statistics method will be added gradually in the future. Group/VarsX/VarsBy/Method/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatDesc can only run successfully after successfully running InitStat and LoadStat functions.

Group/VarsX/VarsBy/Method/Layout/Brightness/Palette parameters can be a character, run `?StatCorr` to see more details.

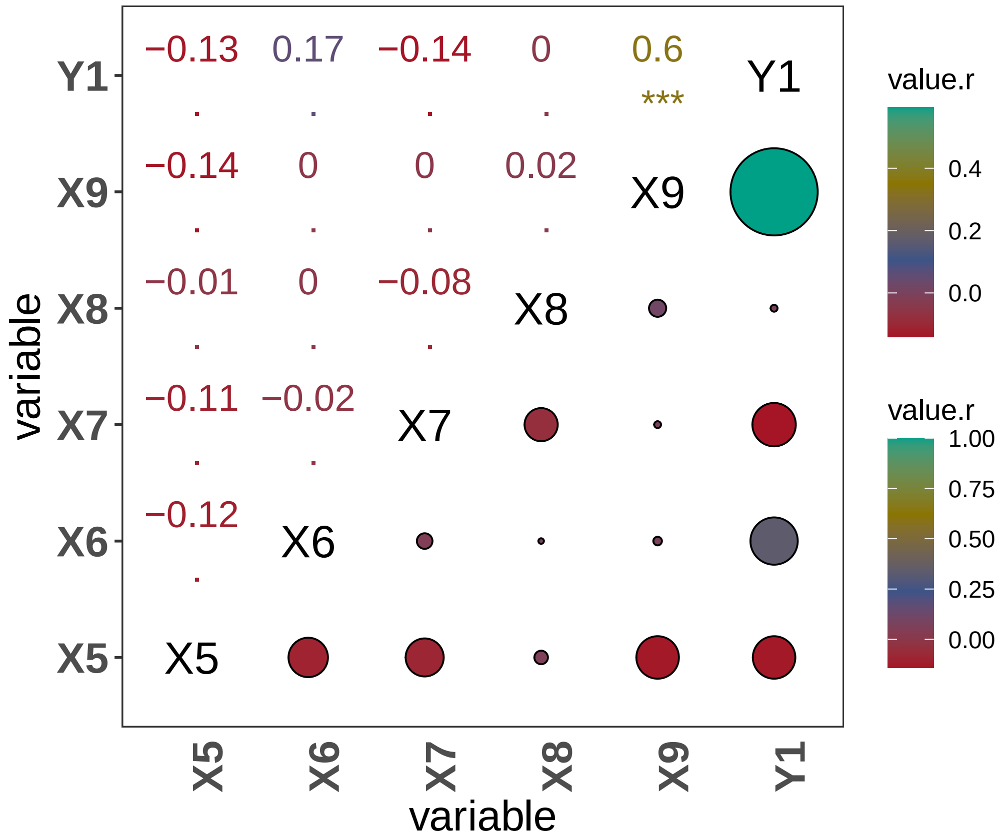
```
res = InitStat()

res1 = LoadStat(PID = res$PID,
                UseExample = "example#1")

res2 = StatCorr(PID = res$PID,
                Group = T,
                VarsX = "X5,X6,X7,X8,X9",
                VarsY = "Y1",
                VarsBy = "Y1",
                Method = "pearson",
                Layout = "bubble",
                Brightness = "dark",
                Palette = "nature")
res2
```

```
## $Corrtable
## $Corrtable$StatCorr_pearson
## # A tibble: 24 x 9
##    Terms Group Vars       X5       X6       X7       X8      X9       Y1
##    <chr> <chr> <chr>   <dbl>    <dbl>    <dbl>    <dbl>   <dbl>    <dbl>
```

```
##   1 r     Train X5     1        0.0303  -0.0578  -0.111    -0.0689 -0.0680
##   2 r     Train X6     0.0303  1        -0.110   -0.00523  0.0151  0.0127
##   3 r     Train X7    -0.0578 -0.110    1       -0.131     0.0345 -0.00585
##   4 r     Train X8    -0.111  -0.00523 -0.131    1        -0.0138  0.132
##   5 r     Train X9    -0.0689  0.0151   0.0345  -0.0138    1       0.626
##   6 r     Train Y1    -0.0680  0.0127  -0.00585  0.132     0.626   1
##   7 P     Train X5    NA       0.717    0.486    0.181     0.406   0.411
##   8 P     Train X6     0.717  NA        0.184    0.950     0.856   0.878
##   9 P     Train X7     0.486   0.184   NA        0.111     0.675   0.943
##  10 P     Train X8     0.181   0.950    0.111   NA         0.867   0.108
## # ... with 14 more rows
##
##
## $CorrPlot
## $CorrPlot$VizStatCorr_Bubble_Train_dark_nature
```




```
##
## $CorrPlot$VizStatCorr_Bubble_Test_dark_nature
```




```
##
##
## $CorrPlot_para
## $CorrPlot_para$VizStatCorr_Bubble_Train_dark_nature
## $CorrPlot_para$VizStatCorr_Bubble_Train_dark_nature$width
## [1] 11.6
##
## $CorrPlot_para$VizStatCorr_Bubble_Train_dark_nature$height
## [1] 11
##
##
## $CorrPlot_para$VizStatCorr_Bubble_Test_dark_nature
## $CorrPlot_para$VizStatCorr_Bubble_Test_dark_nature$width
## [1] 11.6
##
## $CorrPlot_para$VizStatCorr_Bubble_Test_dark_nature$height
## [1] 11
```

After all the analysis is done, please run the "FuncExit()" function to delete the data uploaded to the server.

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```