Multivariate Analysis using R

Singh Richa

2018-12-10

Contents

1 Introduction

The general purpose of all statistical methods – univariate, bivariate, or multivariate is to summarize, reduce, and interpret raw data. Essentially, multivariate analysis is a tool to find patterns and relationships between several variables simultaneously. It lets us predict the effect a change in one variable will have on other variables.This gives multivariate analysis a decisive advantage over other forms of analysis.

The cookbook is all about using different techniques using multivariate Analysis. We will use different types of Multivariate techniques on the IBM-NOAttrition-Hypothetical Dataset and see how different techniques respond to the dataset and which method is best for our dataset i.e exploratory data analysis is done to find out the research answers from the data that are intended. Also, in genral we will conclude which technique is used in what all scenerios.

2 Dataset : IBM-HR-Employee-NoAttrition

There are four measurement scales (or types of data): nominal, ordinal, interval and ratio. These are simply ways to categorize different types of variables.

Nominal

"Nominal" scales could simply be called "labels." They are also known as Qualitative Variables. e.g - What is you Gender? Male and Female are the two labels.

Ordinal

With ordinal scales, it is the order of the values is what's important and significant, but the differences between each one is not really known.Ordinal scales are typically measures of non-numeric concepts like satisfaction, happiness, discomfort, etc.eg- How do you feel today? 1,2,3,4,5 where 1 is very happy and 5 is being unhappy

Qunatitative

Quantitative Variable. Variables that have are measured on a numeric or quantitative scale. Ordinal, interval and ratio scales are quantitative. A country's population, a person's shoe size, or a car's speed are all quantitative variables.

Ratio

Ratios tell us about the order, they tell us the exact value between units and they also have an absolute zero–which allows for a wide range of both descriptive and inferential statistics to be applied.

DataSet Description

The dataset consists of 1233 observations of HR-IBM Employees who didn't leave the office and 32 variables describing them.

Quantitative Variables : Sub,Age,Monthly Income,Daily Rate,Hourly Rate, MonthlyRate, DistanceFrome-Home,PercentSalaryHike,TotalWorkingYears, TrainingTimesLastYear,WorkLifeBalance,WorkLifeBalance,YearsAtCompany,Y YearsWithCurrManager

Ordinal Variables : PerformanceRating, Education, JobInvolvement,Joblevel,StockOptionLevel,EnvironmentSatisfaction, JobSatisfaction, RelationshipSatisfaction

Qualitative variable : Attrition,BusinessTravel,EducationField,Gender,JobRole,MaritalStatus,OverTime

Note: Education: 1 'Below College' 2 'College' 3 'Bachelor' 4 'Master' 5 'Doctor' EnvironmentSatisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High' JobInvolvement: 1 'Low' 2 'Medium' 3 'High' 4 'Very High' JobSatisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High' PerformanceRating: 1 'Low' 2 'Good' 3 'Excellent' 4 'Outstanding' RelationshipSatisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High' WorkLifeBalance: 1 'Bad' 2 'Good' 3 'Better' 4 'Best'

Research Question

How do all 1233 HR-IBM Employees differ on the variables like Gender type vs Department ? Job level vs MonthlyIncome ? etc.Explore important questions such as 'show me a breakdown of distance from home by job role' or 'compare average monthly income by education'. This is a fictional data set created by IBM data scientists.

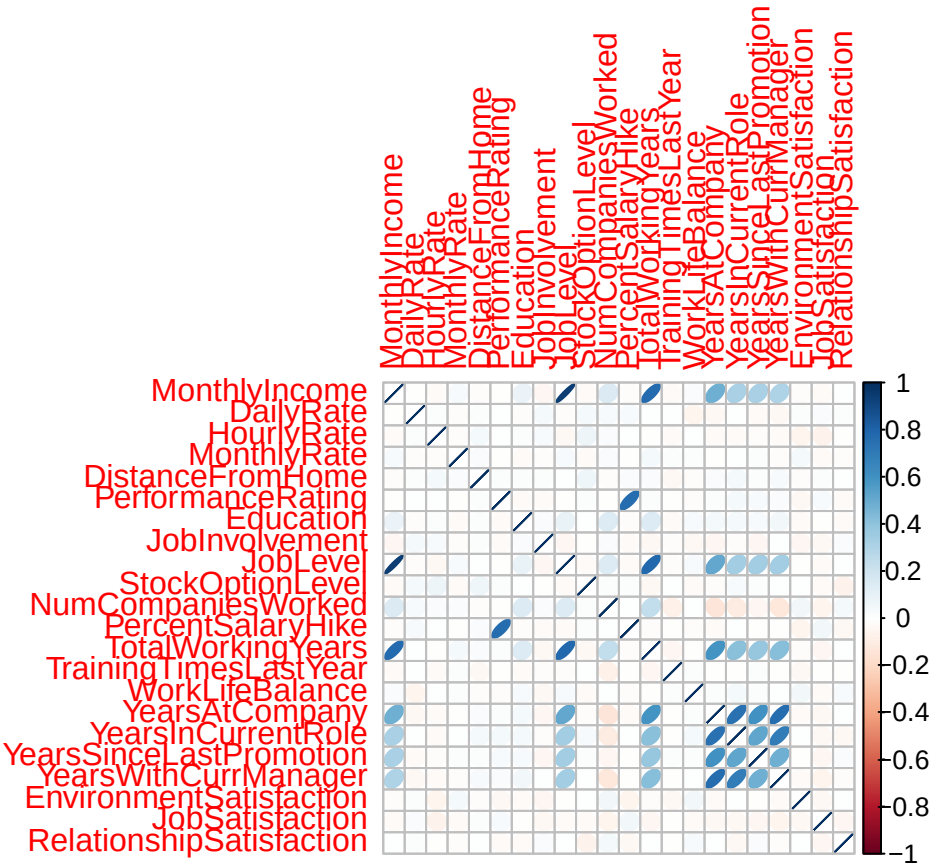
Correlation Plot

In statistics, the correlation coefficient r measures the strength and direction of a linear relationship between two variables on a scatterplot. The value of r is always between +1 and –1.The measure calculates the strength of the relationship between the relative movements of the two variables.

```
datar <- my_data1[,2:23]
datae <- my_data[,5]
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
cor.my_data <- cor(datar)
corrplot(cor.my_data, method = "ellipse")
```



3 PCA

Principal Component Analysis is the analysis of data to identify patterns and finding patterns to reduce the dimensions of the dataset with minimal loss of information. Here, our desired outcome of the principal component analysis is to project a feature space (our dataset consisting of n d-dimensional samples) onto a smaller subspace that represents our data well.PCA gives one map for the rows (called factor scores), and one map for the columns (called loadings).These 2 maps are related, because they both are described by the same components. However, these 2 maps project different kinds of information onto the components, and so they are *interpreted differently*. Factor scores are the coordinates of the row observations. They are interpreted by the distances between them, and their distance from the origin. Loadings describe the column variables. Loadings are interpreted by the angle between them, and their distance from the origin.

`The distance from the origin is important in both maps, because squared distance from the mean is`

We will use the InPosition library(epPCA package) to do PCA: Because each variable is measured on different units, I choose to center and scale the columns. The rows are color-coded by the DESIGN variable, state.division. * `center = TRUE`: substracts the mean from each column * `scale = TRUE`: after centering (or not), scales each column (see the help for different scaling options) * `DESIGN`: colors the observations (rows)

```
library(InPosition)
resPCA <- epPCA(DATA = datar,
                scale = 'SS1', # Make to use 'SS1' rather than TRUE
                DESIGN = datae,
                graphs =  FALSE # TRUE first pass only
)
```

```
testVari    <- data4PCCAR::epVari(resPCA)
resPCA.inf <- InPosition::epPCA.inference.battery(DATA = datar,
                                                  scale = 'SS1', # Make sure to use 'SS1' rather than T
                                                  DESIGN = datae,
                                                  graphs =  FALSE # TRUE first pass only
)
```

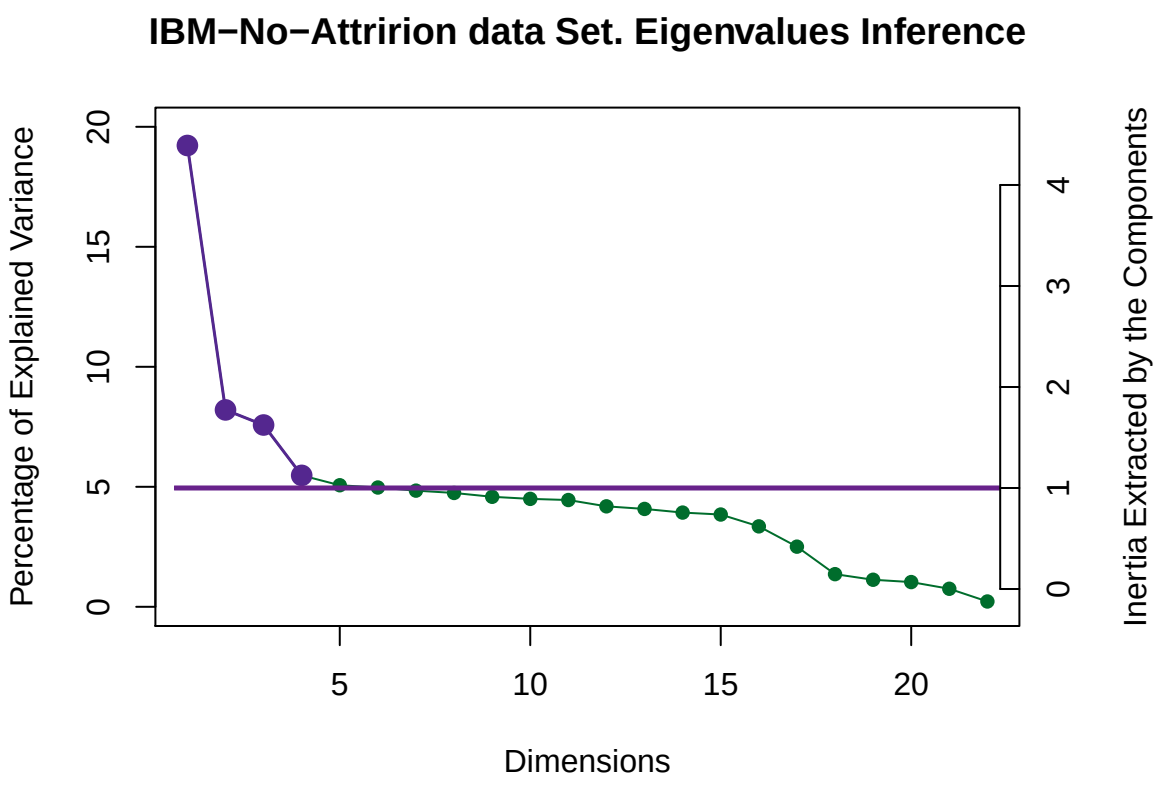
```
## [1] "It is estimated that your iterations will take 0.06 minutes."
## [1] "R is not in interactive() mode. Resample-based tests will be conducted. Please take note of the
## ============================================================================
```

3.1 Scree Plot

A Scree Plot is a simple line segment plot that shows the fraction of total variance in the data as explained or represented by each PC.(`In the PCA literature, the plot is called a 'Scree' Plot because it often looks like a 'scree' slope, where rocks have fallen down and accumulated on the side of a mountain.`)The scree plot shows the eigenvalues, the amount of information on each component. The number of components (the dimensionality of the factor space) is min(nrow(DATA), ncol(DATA)) minus 1. Here, min(1233,31)-1 give 30 components. The scree plot is used to determine how many of the components should be interpreted.

- `plot` draws the line that connects all data points by `type = "l"`
- The first `points` function draws round purple dots.
- The second `points` function draws black circles around the dots (just to make it prettier).

```
PlotScree(ev = resPCA$ExPosition.Data$eigs,
          p.ev =  resPCA.inf$Inference.Data$components$p.vals,
          title = 'IBM-No-Attririon data Set. Eigenvalues Inference',
          plotKaiser = TRUE
)
```

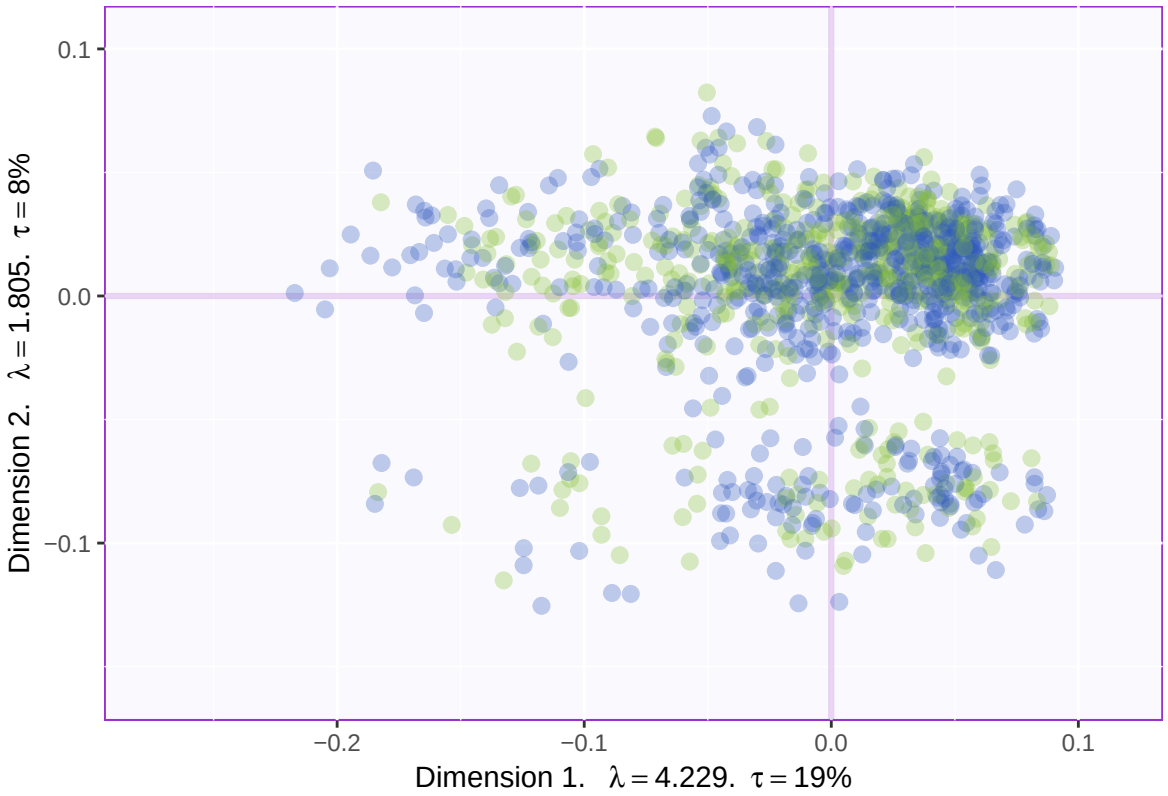


The top 4 compnents can be analyzed.

3.2 Factor Scores

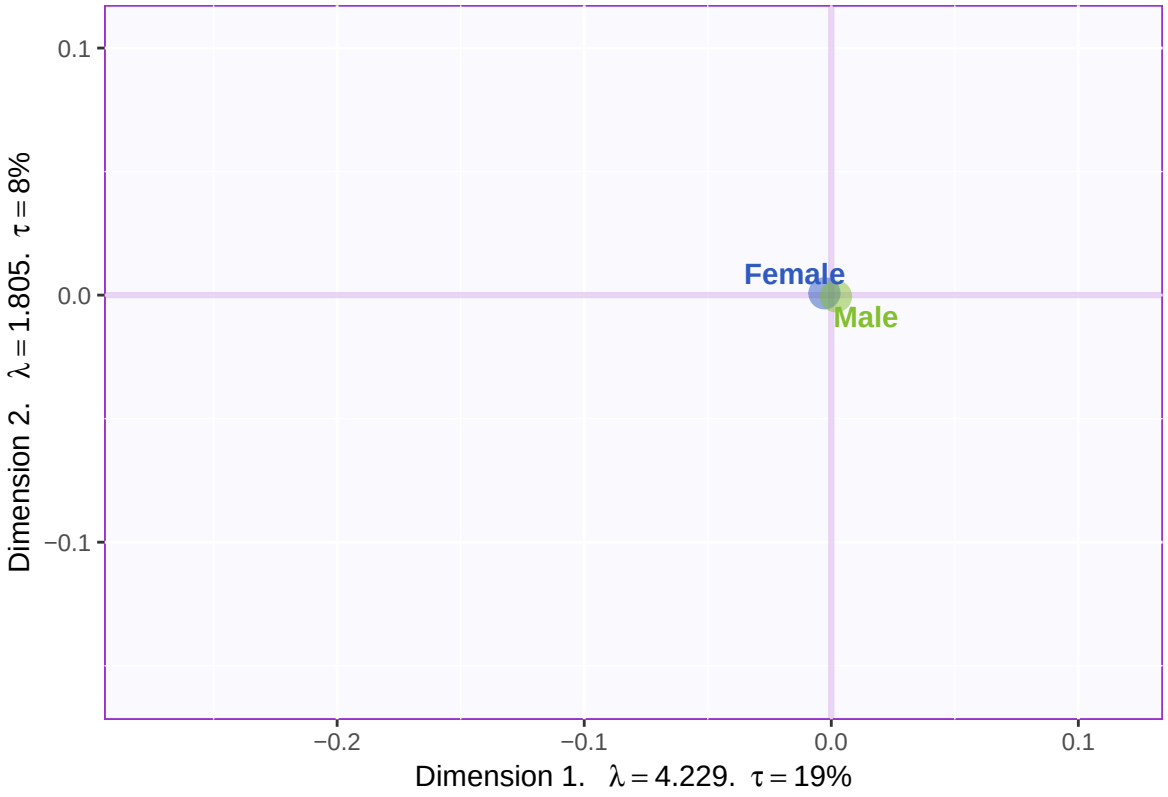
Gender Variable

```
resPCA0 <- epPCA(DATA = datar,
                 scale = 'SS1', # Make to use 'SS1' rather than TRUE
                 DESIGN = my_data$Gender,
                 graphs =  FALSE # TRUE first pass only
)

baseMap.i0 <- PTCA4CATA::createFactorMap(resPCA0$ExPosition.Data$fi,
                                         col.points   = resPCA0$Plotting.Data$fi.col,
                                         alpha.points =  .3)
label4Map0 <- createxyLabels.gen(1,2,
                                 lambda = resPCA0$ExPosition.Data$eigs,
                                 tau = resPCA0$ExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i0 <- baseMap.i0$zeMap_background + baseMap.i0$zeMap_dots + label4Map0
#-----------------------------------------------------------------------
print(aggMap.i0)
```



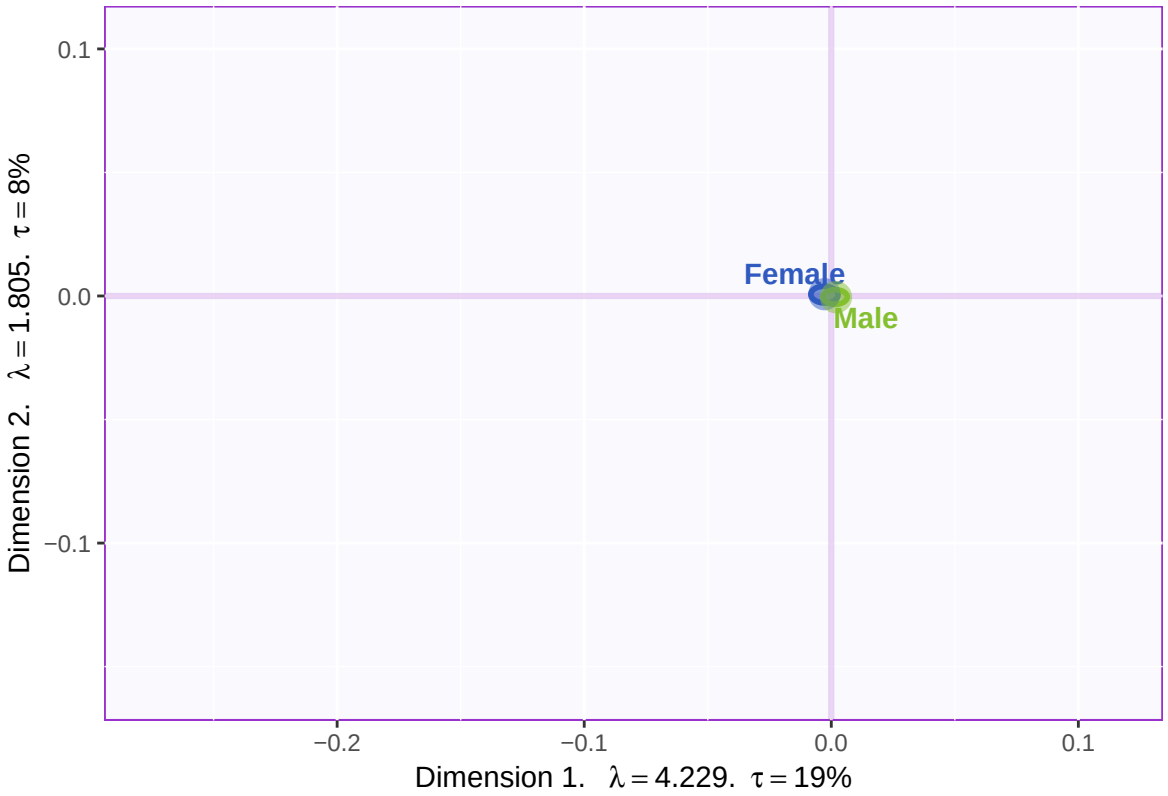

```
col4data0 <- resPCA0$Plotting.Data$fi.col
col4Means0 <- unique(col4data0)
dataMeans0 <- getMeans(resPCA0$ExPosition.Data$fi, my_data$Gender)

MapGroup0    <- PTCA4CATA::createFactorMap(dataMeans0,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means0,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means0,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                              col.background = adjustcolor("lavender",
                                                           alpha.f = 0.2),
                              force = 1, segment.size = 0)

aggMap.i.withMeans0 <- baseMap.i0$zeMap_background  +
  MapGroup0$zeMap_dots + MapGroup0$zeMap_text  + label4Map0
#-------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans0)
```



C.I of the mean

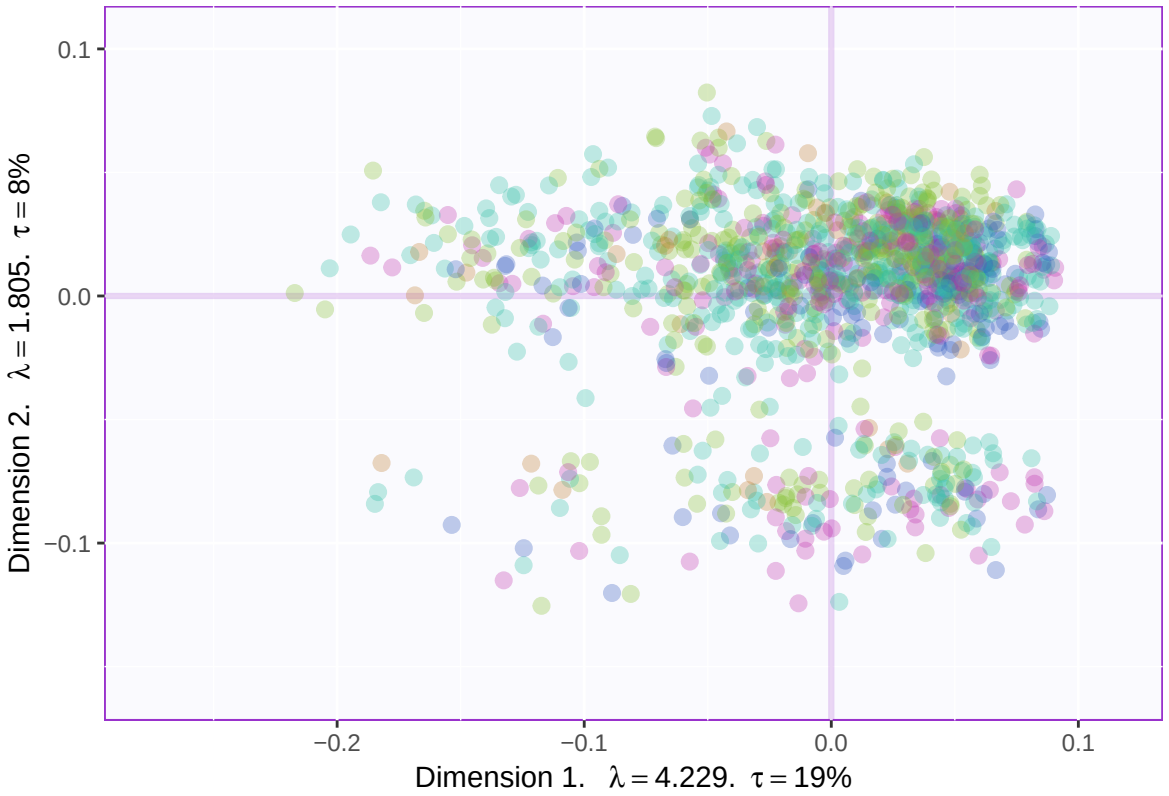
```
BootCube.Gr0 <- PTCA4CATA::Boot4Mean(resPCA0$ExPosition.Data$fi, design = my_data$Gender,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli0 <- MakeCIEllipses(BootCube.Gr0$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means0,
                   p.level = .95
)


aggMap.i.withCI0 <-  baseMap.i0$zeMap_background +  GraphElli0 + MapGroup0$zeMap_text + MapGroup0$zeMap_

print(aggMap.i.withCI0)
```



From the above plot we can see that no inference could be concluded as Males and Females almost overlap each other.

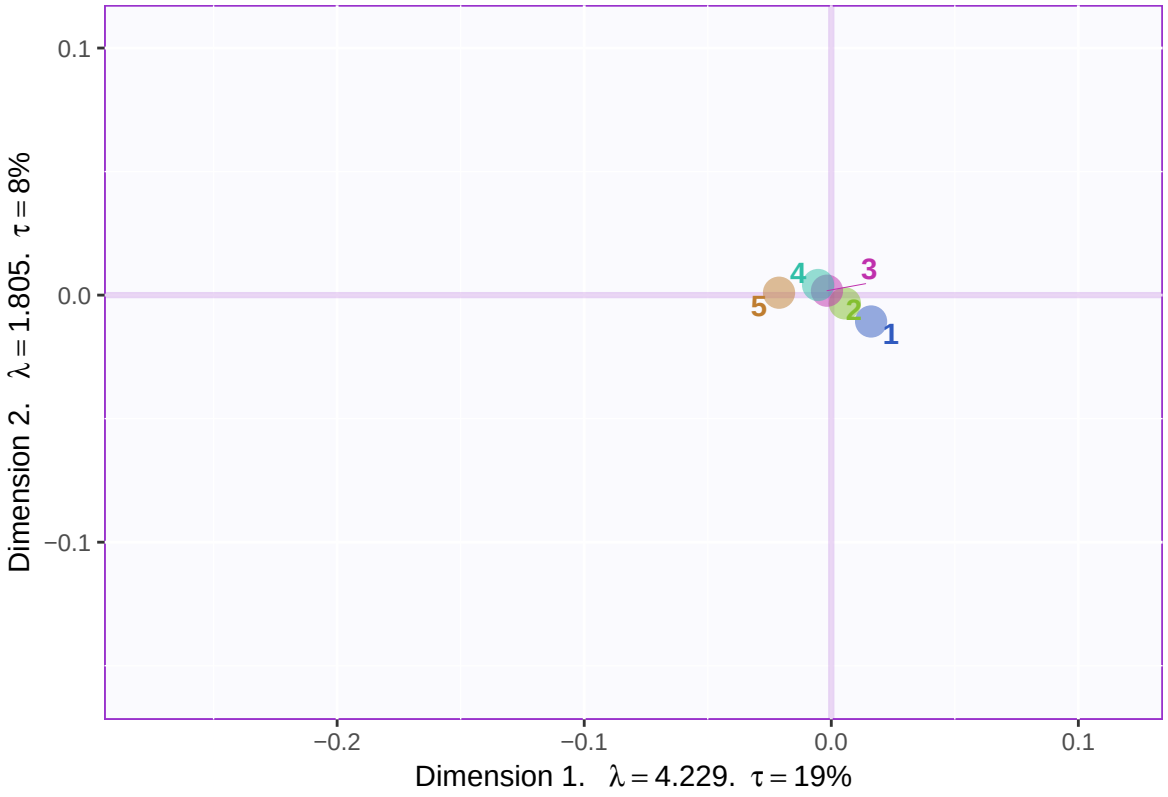
Education

```
resPCA1 <- epPCA(DATA = datar,
                 scale = 'SS1', # Make to use 'SS1' rather than TRUE
                 DESIGN = my_data$Education,
                 graphs =  FALSE # TRUE first pass only
)

baseMap.i1 <- PTCA4CATA::createFactorMap(resPCA1$ExPosition.Data$fi,
                                         col.points   = resPCA1$Plotting.Data$fi.col,
                                         alpha.points =  .3)
label4Map <- createxyLabels.gen(1,2,
                                lambda = resPCA1$ExPosition.Data$eigs,
                                tau = resPCA1$ExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i1 <- baseMap.i1$zeMap_background + baseMap.i1$zeMap_dots + label4Map
#-----------------------------------------------------------------------
print(aggMap.i1)
```

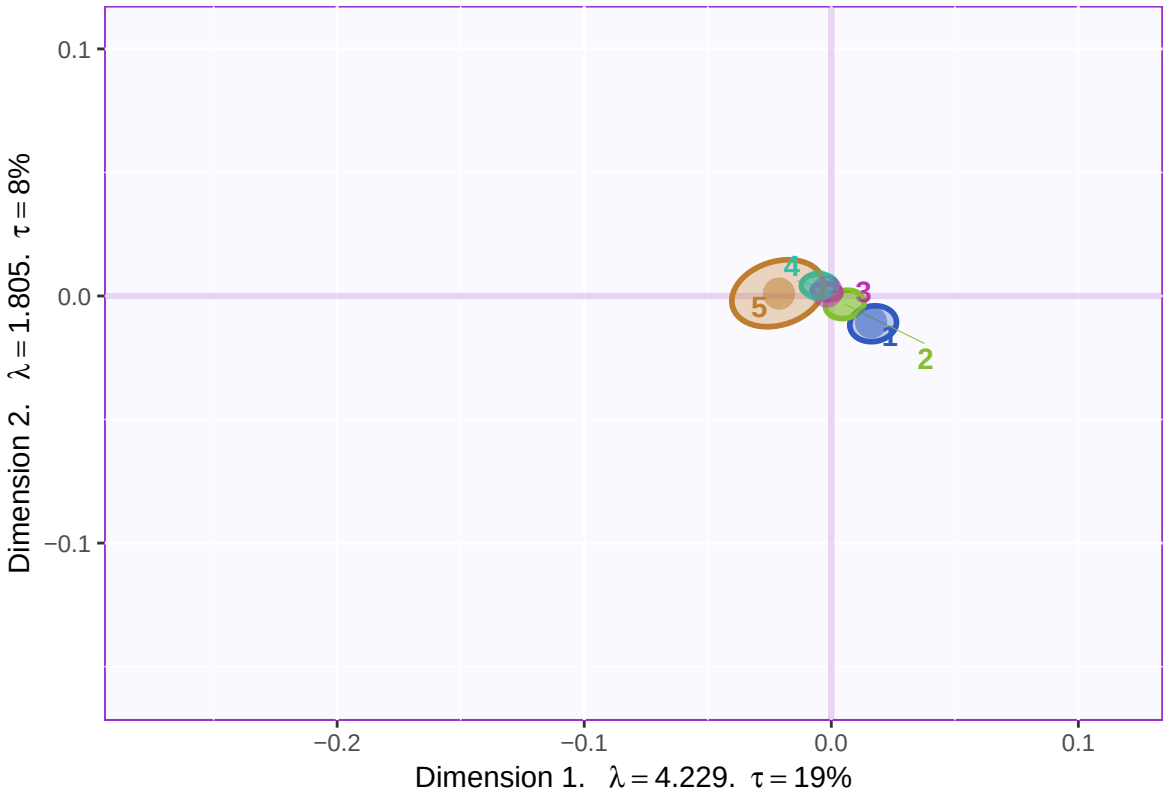



```
col4data1 <- resPCA1$Plotting.Data$fi.col
col4Means1 <- unique(col4data1)
dataMeans1 <- getMeans(resPCA1$ExPosition.Data$fi, my_data$Education)

MapGroup1    <- PTCA4CATA::createFactorMap(dataMeans1,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means1,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means1,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans1 <- baseMap.i1$zeMap_background  +
  MapGroup1$zeMap_dots + MapGroup1$zeMap_text + label4Map
#------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans1)
```




```
BootCube.Gr1 <- PTCA4CATA::Boot4Mean(resPCA1$ExPosition.Data$fi, design = my_data$Education,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli1 <- MakeCIEllipses(BootCube.Gr1$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means1,
                   p.level = .95
)


aggMap.i.withCI1 <- baseMap.i1$zeMap_background +  GraphElli1 + MapGroup1$zeMap_text + MapGroup1$zeMap_

print(aggMap.i.withCI1)
```

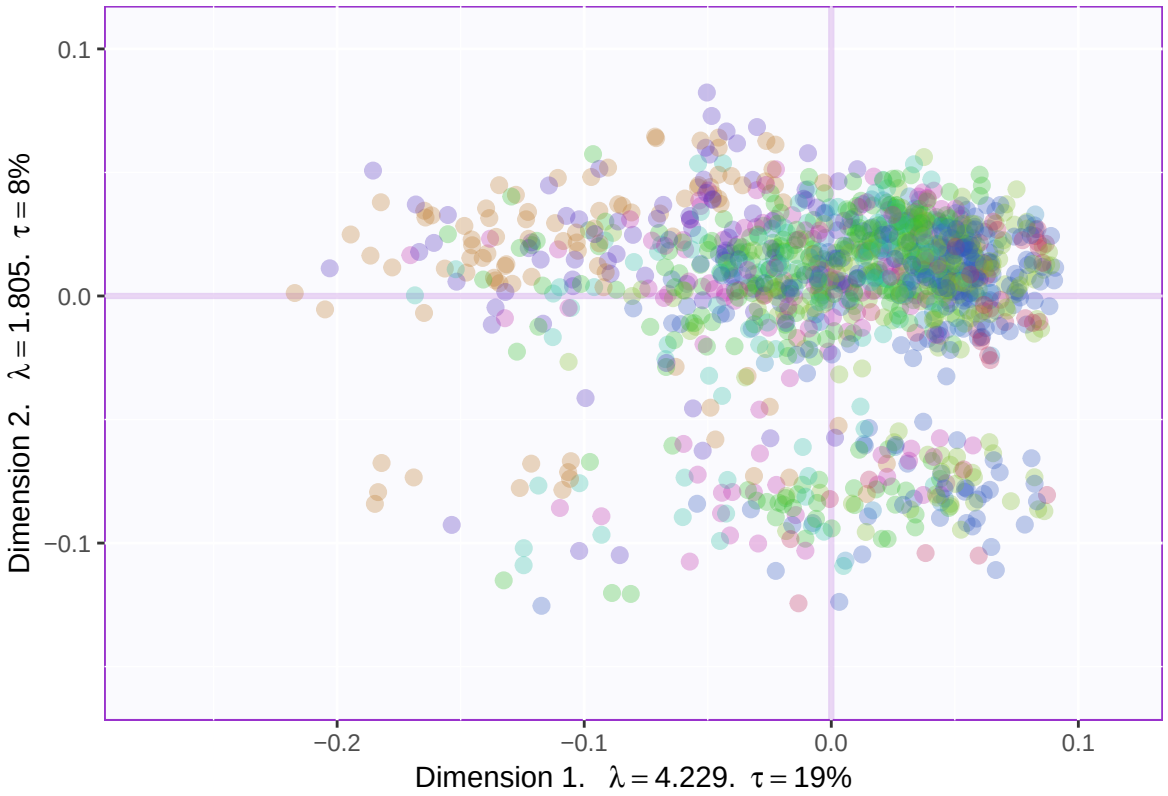


Education: 1 'Below College' 2 'College' 3 'Bachelor' 4 'Master' 5 'Doctor'

So, we can say that the 1st dimension divides the Below college, College VS Mater's and PhD candidates.

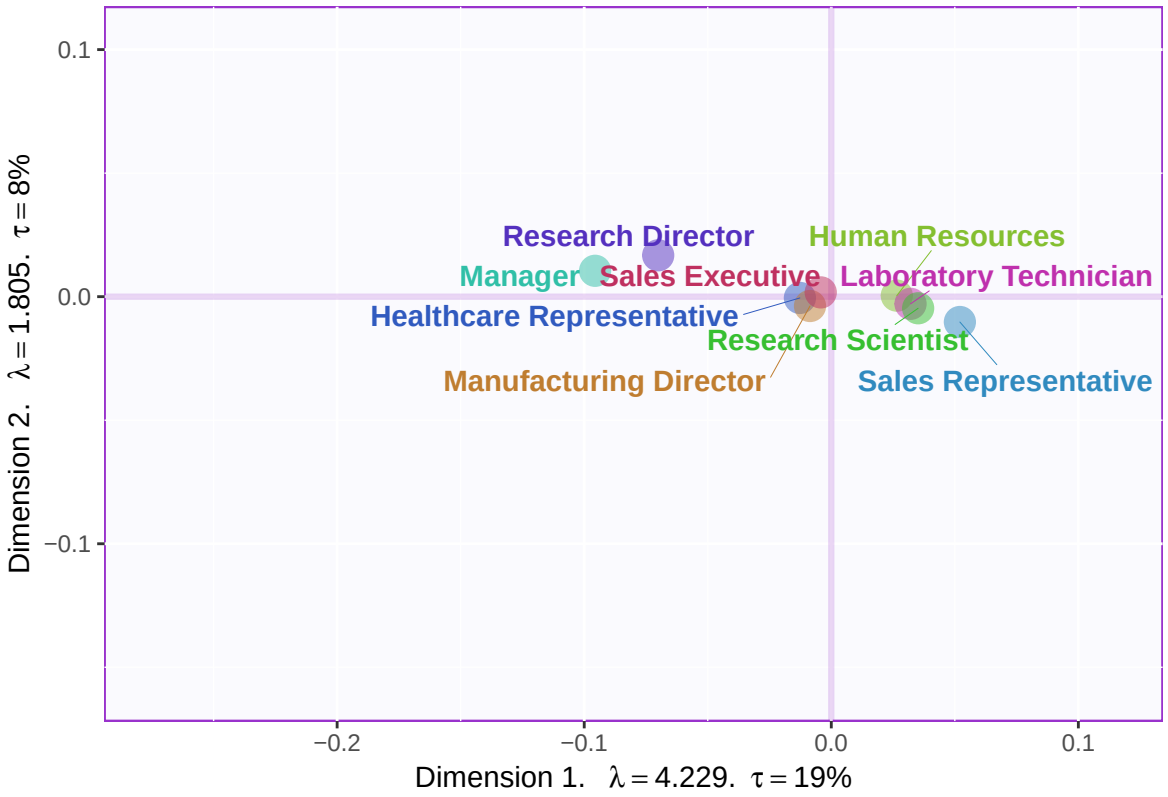
Job Role

```
resPCA2 <- epPCA(DATA = datar,
                 scale = 'SS1', # Make to use 'SS1' rather than TRUE
                 DESIGN = my_data$JobRole,
                 graphs =  FALSE # TRUE first pass only
)

baseMap.i12 <- PTCA4CATA::createFactorMap(resPCA2$ExPosition.Data$fi,
                                          col.points   = resPCA2$Plotting.Data$fi.col,
                                          alpha.points =  .3)
label4Map2 <- createxyLabels.gen(1,2,
                                 lambda = resPCA2$ExPosition.Data$eigs,
                                 tau = resPCA2$ExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i12 <- baseMap.i12$zeMap_background + baseMap.i12$zeMap_dots + label4Map2
#-----------------------------------------------------------------------
print(aggMap.i12)
```

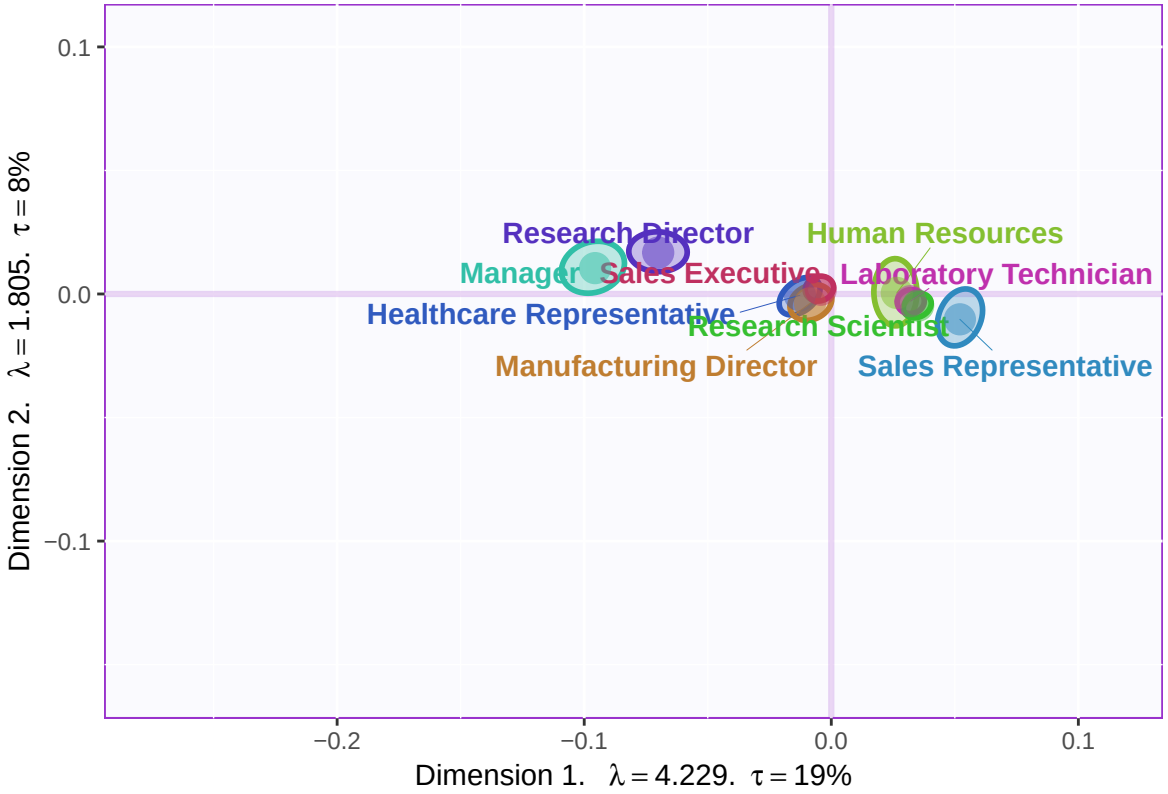



```
col4data2 <- resPCA2$Plotting.Data$fi.col
col4Means2 <- unique(col4data2)
dataMeans2 <- getMeans(resPCA2$ExPosition.Data$fi, my_data$JobRole)

MapGroup2    <- PTCA4CATA::createFactorMap(dataMeans2,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means2,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means2,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans12 <- baseMap.i12$zeMap_background  +
  MapGroup2$zeMap_dots + MapGroup2$zeMap_text + label4Map2
#------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans12)
```



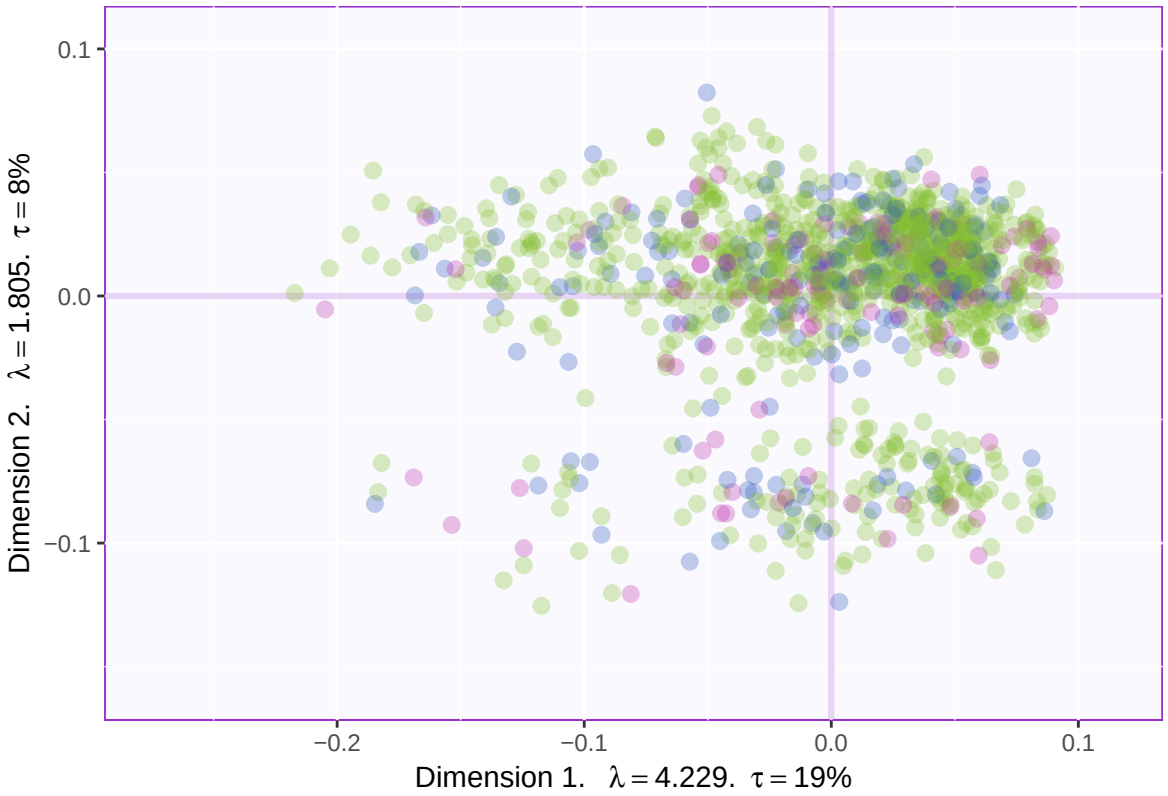

```
BootCube.Gr2 <- PTCA4CATA::Boot4Mean(resPCA2$ExPosition.Data$fi, design = my_data$JobRole,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli2 <- MakeCIEllipses(BootCube.Gr2$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means2,
                   p.level = .95
)


aggMap.i.withCI2 <- baseMap.i12$zeMap_background +  GraphElli2 + MapGroup2$zeMap_text + MapGroup2$zeMap_

print(aggMap.i.withCI2)
```



From the above graph we can say that the first dimension divides the Sales Representative, Research Scientists, Laboratory Technicians VS Managers, Research Directors and Sales Executives.

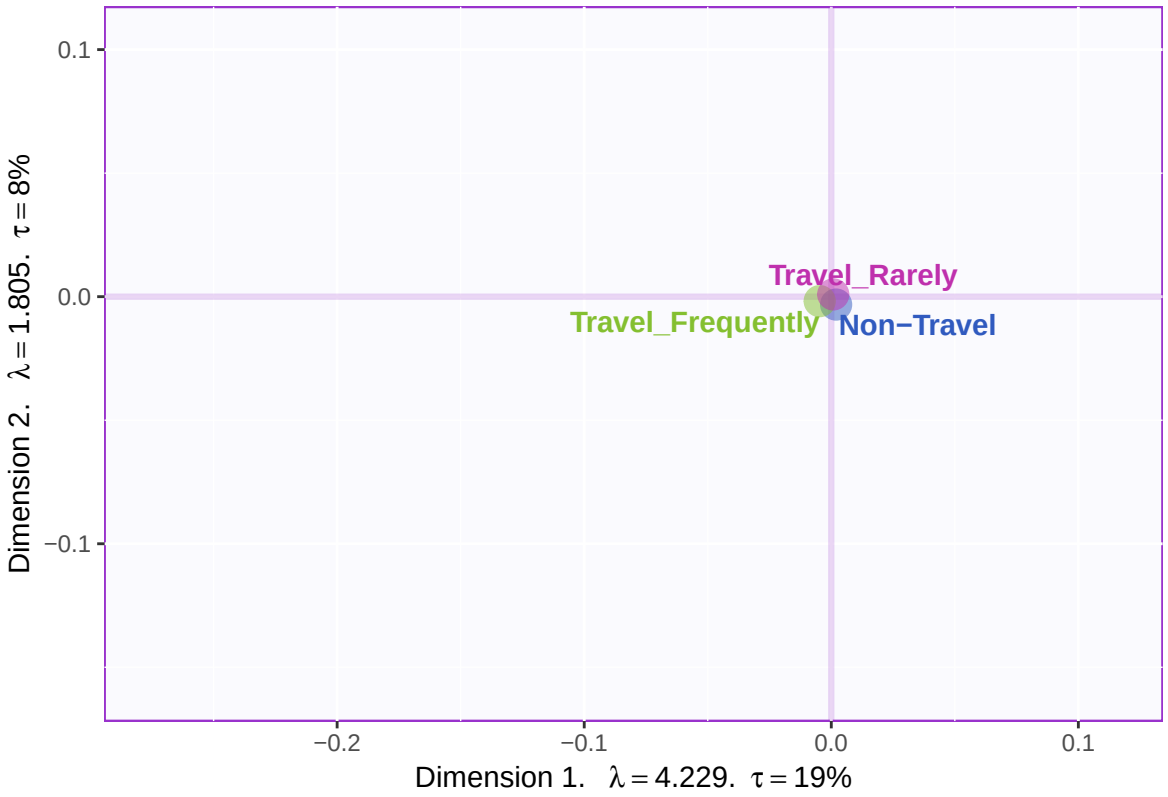
Business Travel

```
resPCA4 <- epPCA(DATA = datar,
                 scale = 'SS1', # Make to use 'SS1' rather than TRUE
                 DESIGN = my_data$BusinessTravel,
                 graphs =  FALSE # TRUE first pass only
)

baseMap.i4 <- PTCA4CATA::createFactorMap(resPCA4$ExPosition.Data$fi,
                                         col.points   = resPCA4$Plotting.Data$fi.col,
                                         alpha.points =  .3)
label4Map4 <- createxyLabels.gen(1,2,
                                 lambda = resPCA4$ExPosition.Data$eigs,
                                 tau = resPCA4$ExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i4 <- baseMap.i4$zeMap_background + baseMap.i4$zeMap_dots + label4Map4
#------------------------------------------------------------------------
print(aggMap.i4)
```

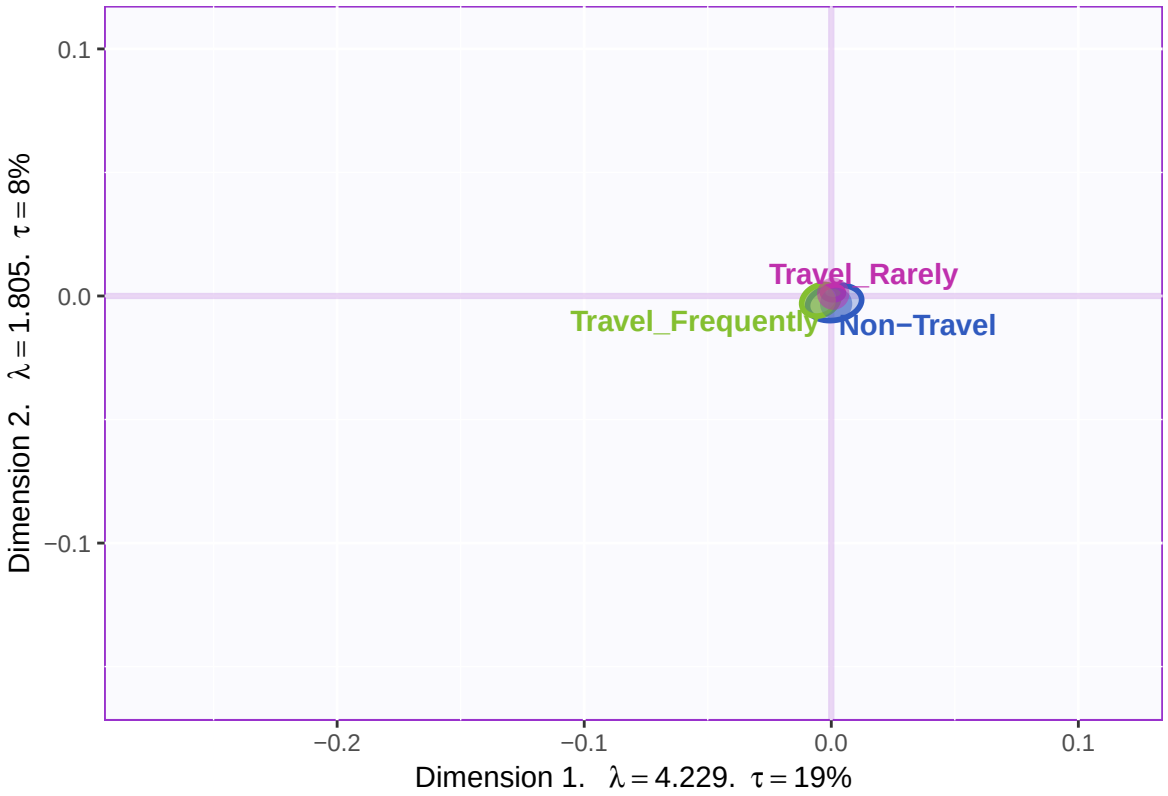



```
col4data4 <- resPCA4$Plotting.Data$fi.col
col4Means4 <- unique(col4data4)
dataMeans4 <- getMeans(resPCA4$ExPosition.Data$fi, my_data$BusinessTravel)

MapGroup4    <- PTCA4CATA::createFactorMap(dataMeans4,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means4,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means4,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans14 <- baseMap.i4$zeMap_background  +
  MapGroup4$zeMap_dots + MapGroup4$zeMap_text + label4Map4
#------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans14)
```



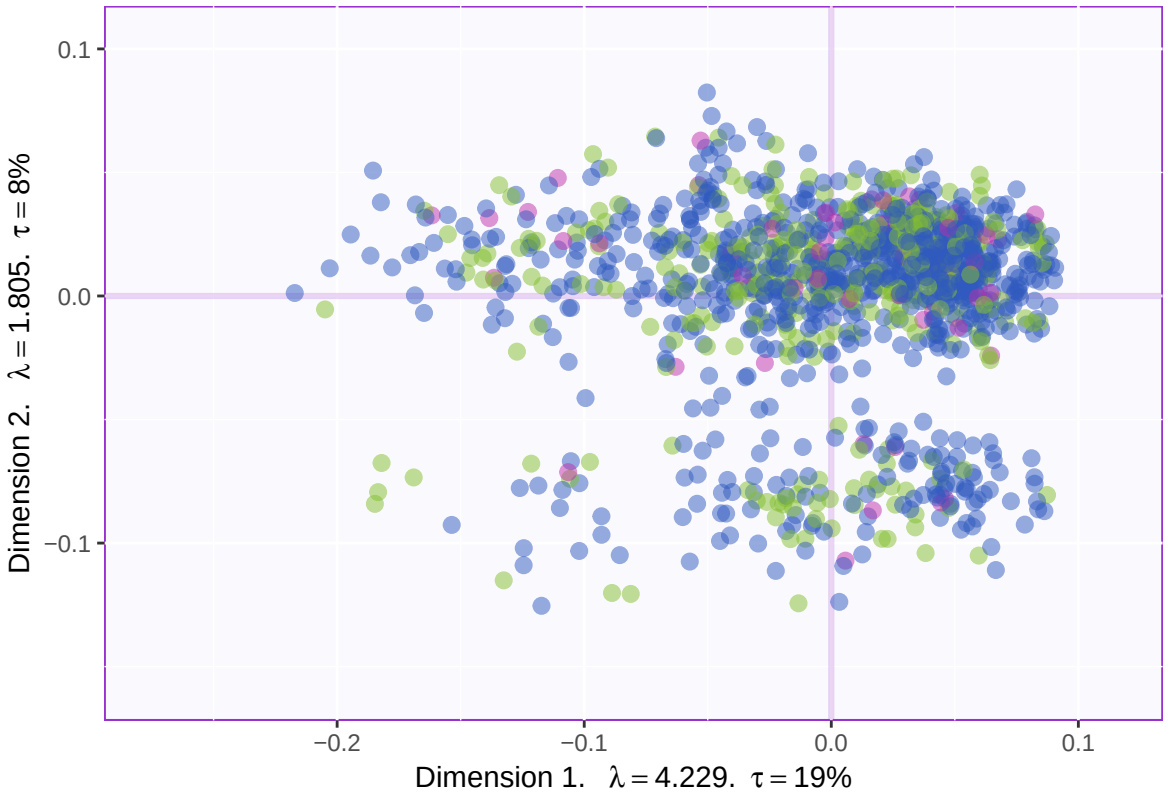

```
BootCube.Gr4 <- PTCA4CATA::Boot4Mean(resPCA4$ExPosition.Data$fi, design = my_data$BusinessTravel,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli4 <- MakeCIEllipses(BootCube.Gr4$BootCube[,1:2,],
                  names.of.factors = c("Dimension 1","Dimension 2"),
                  col = col4Means4,
                  p.level = .95
)


aggMap.i.withCI4 <- baseMap.i4$zeMap_background +  GraphElli4 + MapGroup4$zeMap_text + MapGroup4$zeMap_

print(aggMap.i.withCI4)
```



No significant inference can be observed here.

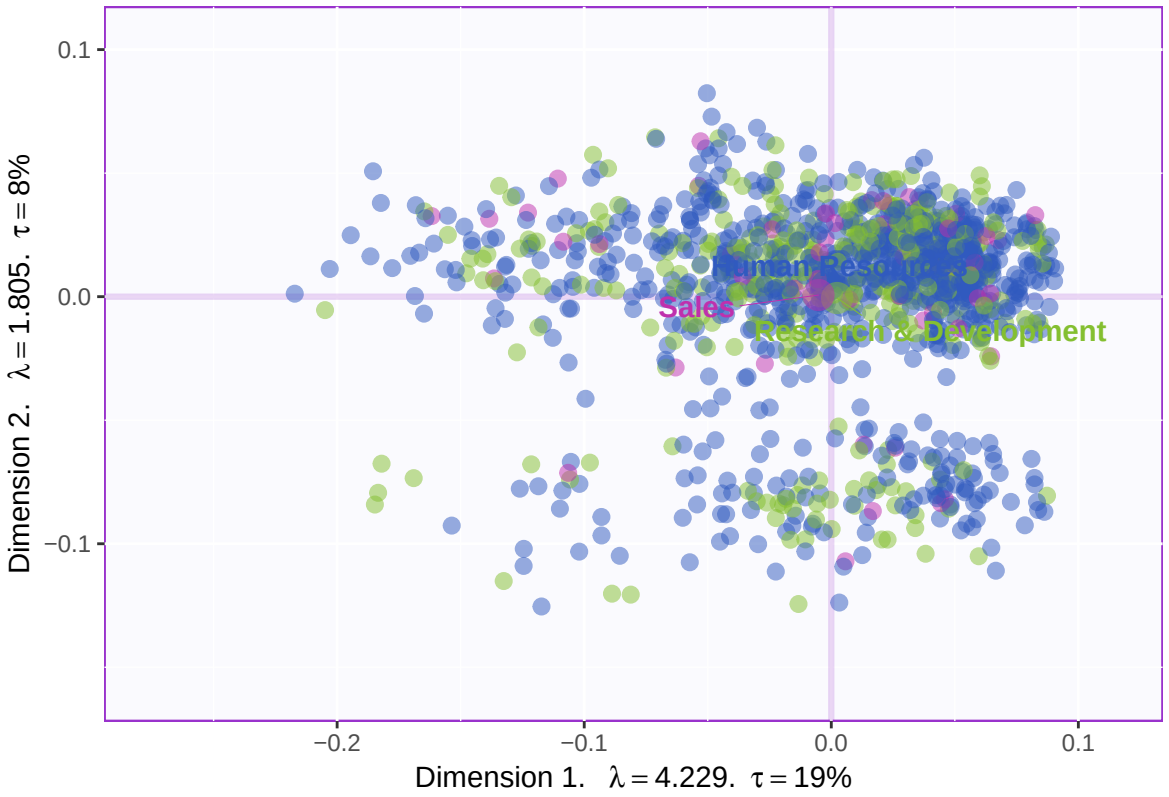
Department

```
# a graph of the observations
# a graph of the observations

my_data1.Imap <- PTCA4CATA::createFactorMap(
  resPCA$ExPosition.Data$fi,
  col.points = resPCA$Plotting.Data$fi.col,
  display.labels = FALSE,
  alpha.points = .5
)
label4Map <- createxyLabels.gen(1,2,
                                 lambda =resPCA$ExPosition.Data$eigs,
                                 tau = resPCA$ExPosition.Data$t)
a002.Map.I <- my_data1.Imap$zeMap + label4Map
print(a002.Map.I)
```

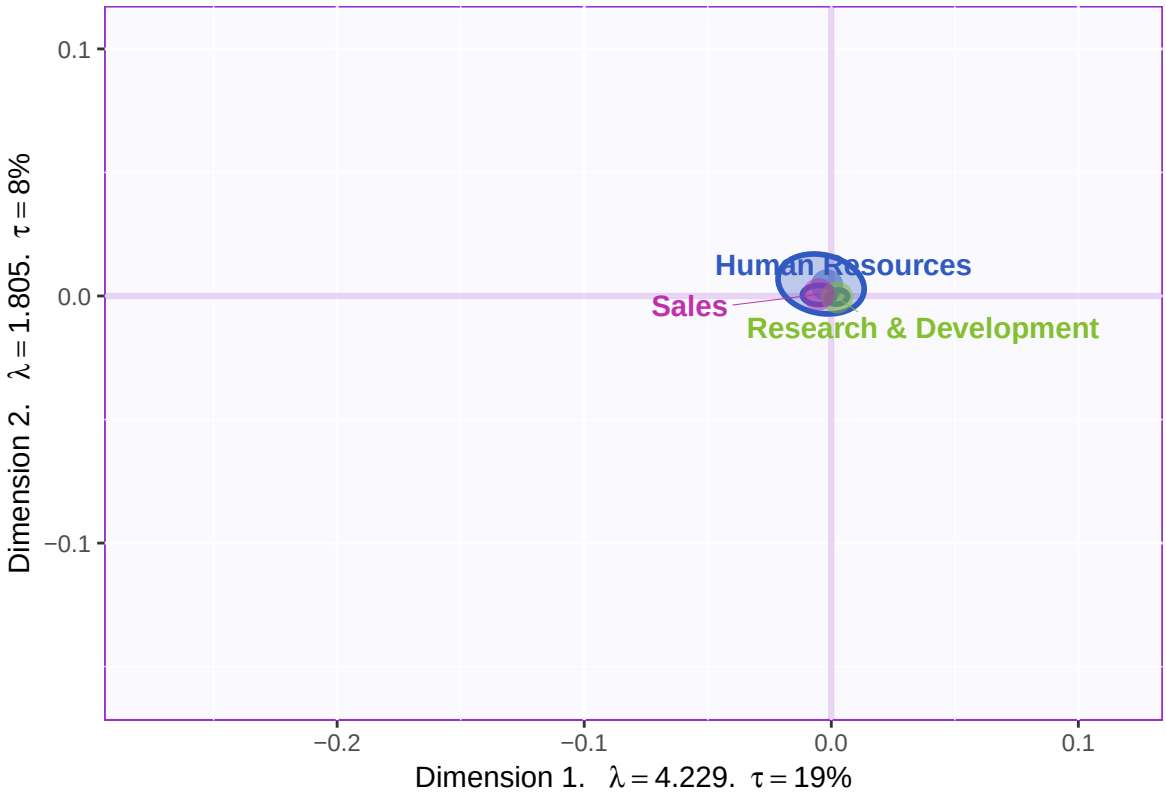



```
col4data3 <- resPCA$Plotting.Data$fi.col
col4Means3 <- unique(col4data3)
dataMeans3 <- getMeans(resPCA$ExPosition.Data$fi, my_data$Department)

MapGroup3    <- PTCA4CATA::createFactorMap(dataMeans3,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means3,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means3,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans13 <- a002.Map.I  +
  MapGroup3$zeMap_dots + MapGroup3$zeMap_text
#------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans13)
```




```
BootCube.Gr3 <- PTCA4CATA::Boot4Mean(resPCA$ExPosition.Data$fi, design = my_data$Department,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli3 <- MakeCIEllipses(BootCube.Gr3$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means2,
                   p.level = .95
)


aggMap.i.withCI3 <- my_data1.Imap$zeMap_background +  GraphElli3 + MapGroup3$zeMap_text + MapGroup3$zeMa

print(aggMap.i.withCI3)
```



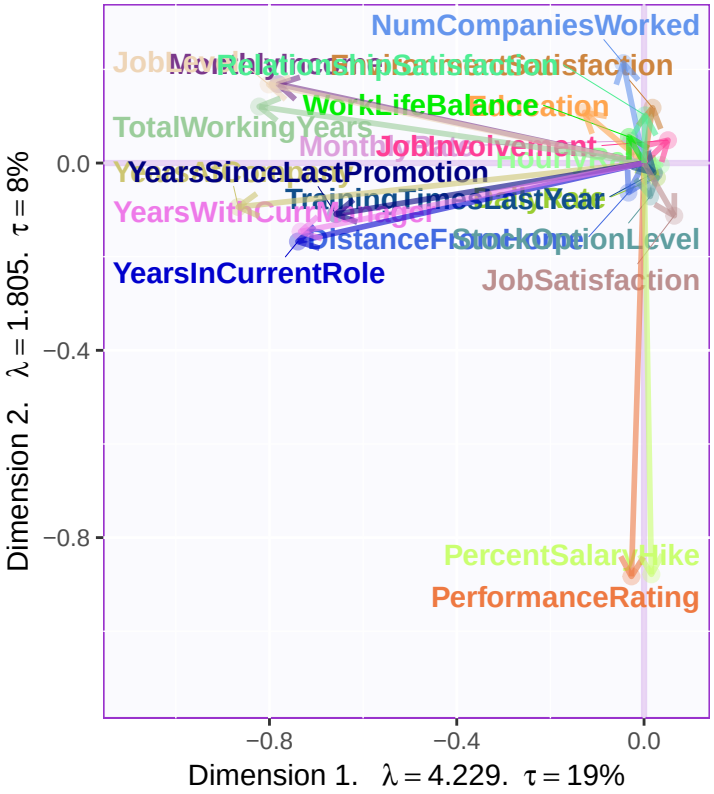
No significant inference can be observed here.

3.3 Loadings

Correlation Circle

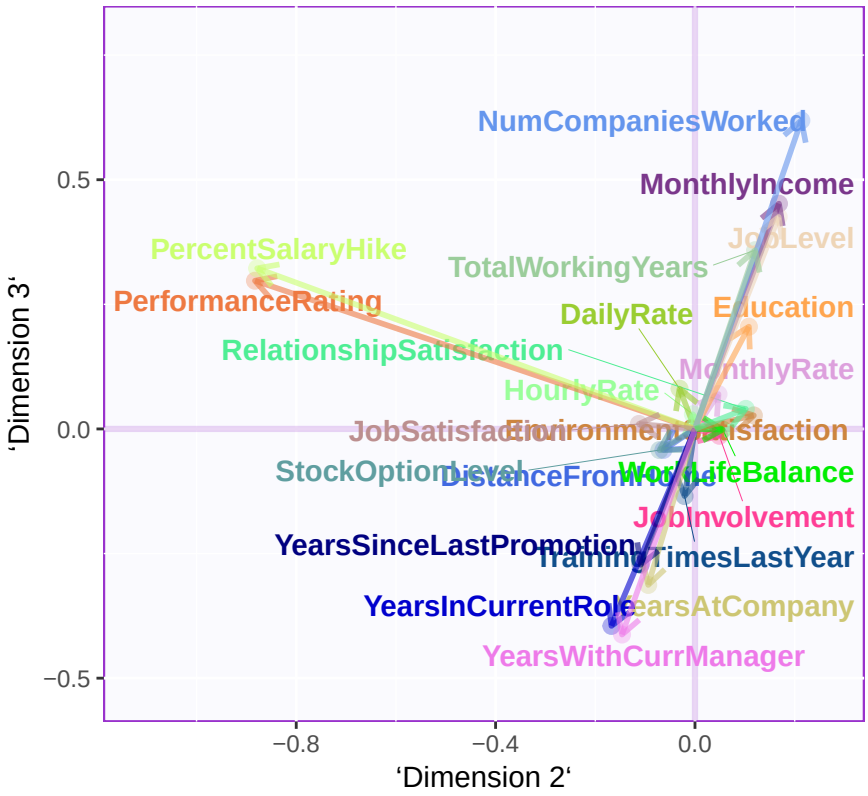
```
correlationCircle <- correlationPlotter(data_matrix = my_data1 , factor_scores =resPCA$ExPosition.Data$
```

```
col4J.ibm <- prettyGraphsColorSelection(NCOL(datar))
baseMap.j <- PTCA4CATA::createFactorMap(resPCA$ExPosition.Data$fj,col.points = col4J.ibm,
                                        col.labels = col4J.ibm ,
                                        alpha.points =  .3)
# arrows
zeArrows <- addArrows(resPCA$ExPosition.Data$fj , col = col4J.ibm)
# A graph for the J-set
b000.aggMap.j <- baseMap.j$zeMap_background + # background layer
  baseMap.j$zeMap_dots + baseMap.j$zeMap_text +  # dots & labels
  label4Map + zeArrows
print(b000.aggMap.j)
```



From the loadings graoh we can say that for dimension 1 Monthly Income, Years of Experience,Total working years are the important variables while on dimension 2 Performance Rating and Percentage SalaryHike are the important variables.

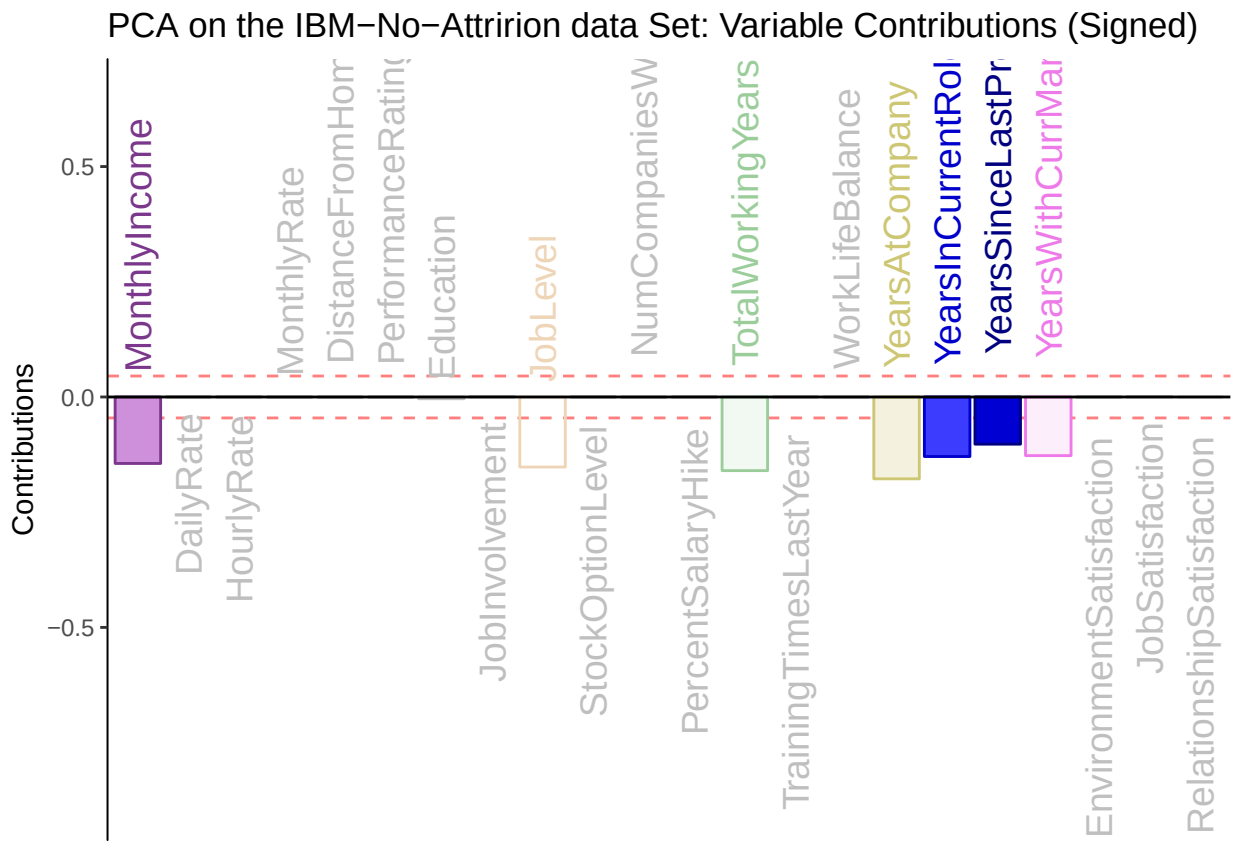
```
baseMap.j <- PTCA4CATA::createFactorMap(resPCA$ExPosition.Data$fj,col.points = col4J.ibm,
                                        col.labels = col4J.ibm , axis1 = 2, axis2 = 3,
                                        alpha.points =  .3)
# arrows
zeArrows1 <- addArrows(resPCA$ExPosition.Data$fj , col = col4J.ibm, axis1 = 2, axis2 = 3)
# A graph for the J-set
b000.aggMap.j <- baseMap.j$zeMap_background + # background layer
  baseMap.j$zeMap_dots + baseMap.j$zeMap_text +  zeArrows1
print(b000.aggMap.j)
```



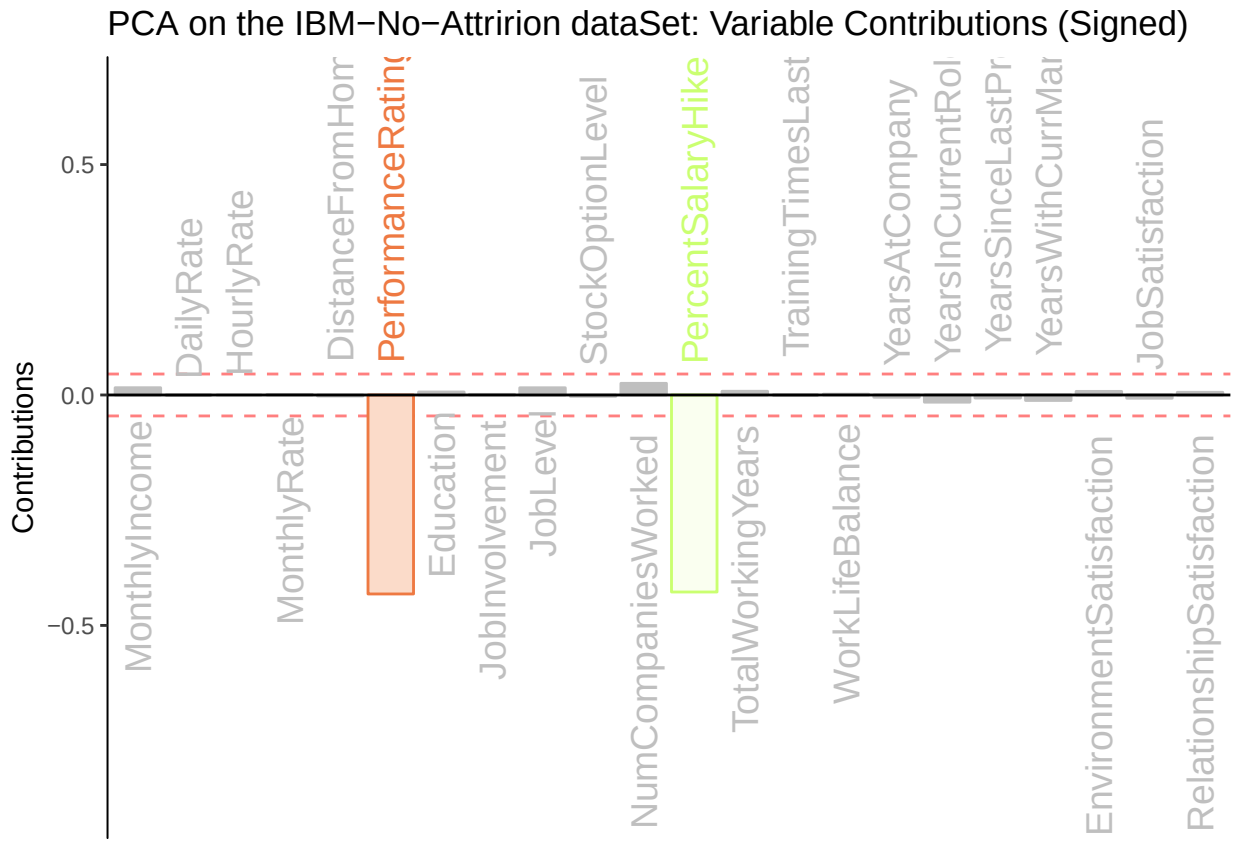
After varimax rotation we can say that

3.4 Contribution

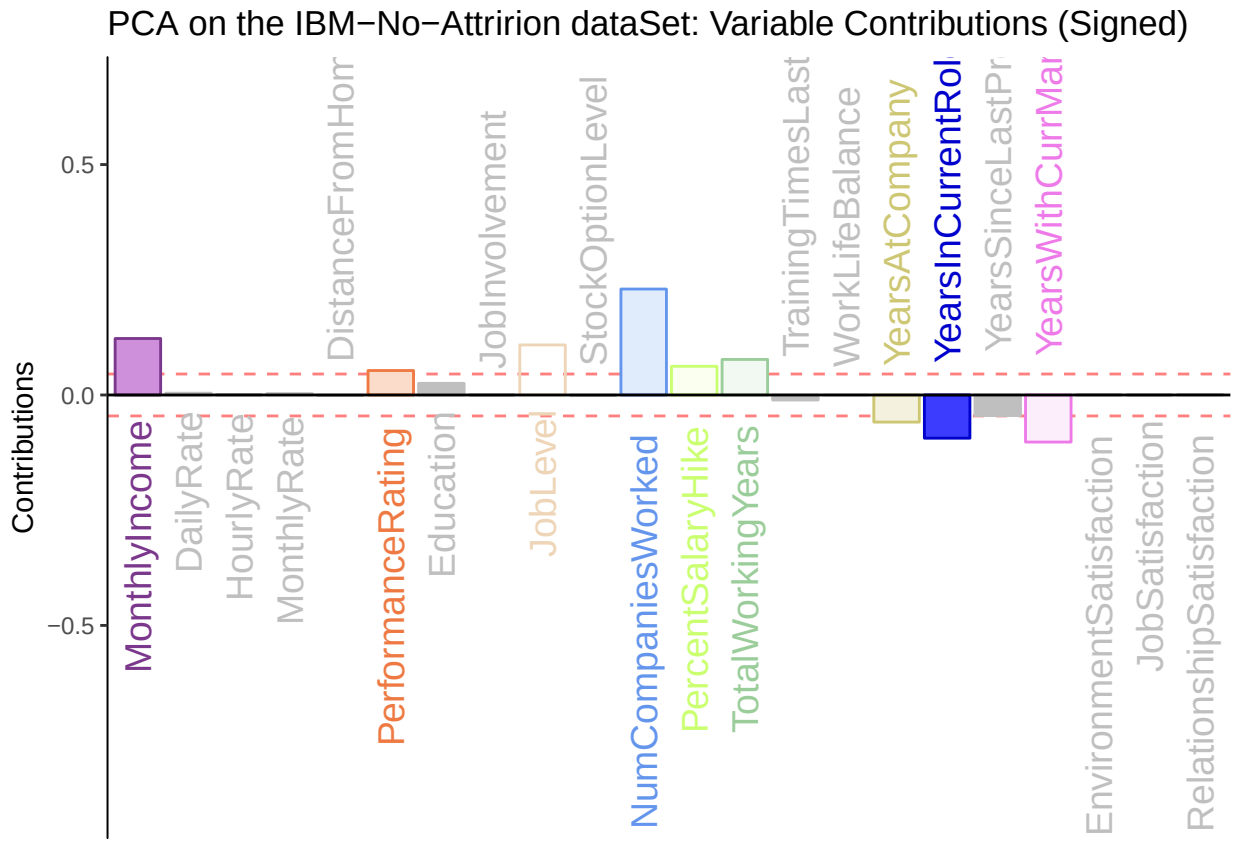
```
signed.ctrJ <- resPCA$ExPosition.Data$cj * sign(resPCA$ExPosition.Data$fj)
b003.ctrJ.s.1 <- PrettyBarPlot2(signed.ctrJ[,1],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                                color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'PCA on the IBM-No-Attririon data Set: Variable Contributions (S
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ))
)
print(b003.ctrJ.s.1)
```




```
b004.ctrJ.s.2 <- PrettyBarPlot2(signed.ctrJ[,2],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                                color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'PCA on the IBM-No-Attririon dataSet: Variable Contributions (Si
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ))
)
print(b004.ctrJ.s.2)
```




```
b004.ctrJ.s.3 <- PrettyBarPlot2(signed.ctrJ[,3],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                                color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'PCA on the IBM-No-Attririon dataSet: Variable Contributions (Si
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ))
)
print(b004.ctrJ.s.3)
```

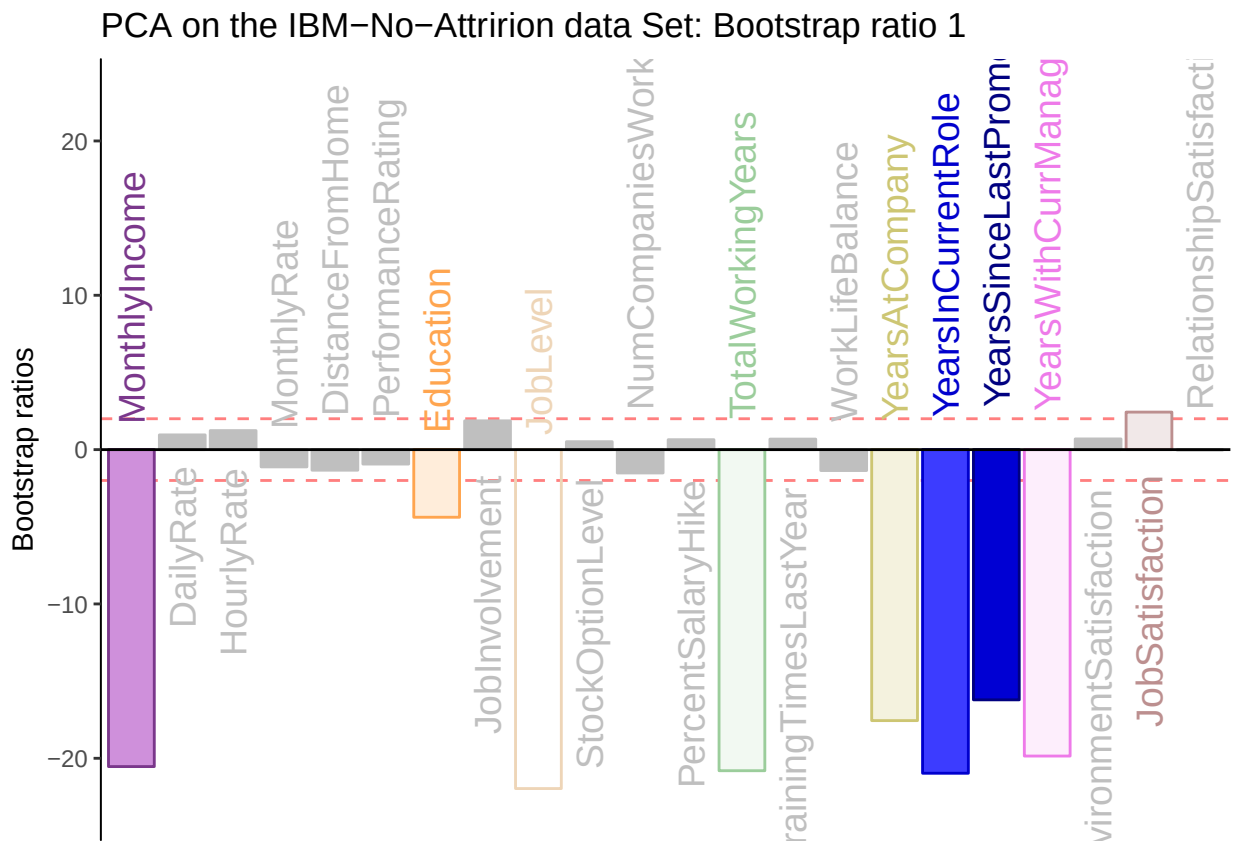


From the contribution graph it is clear that on the first component Monthly Income, JobLevel , TotalWorkingYears , YearsAtCompany , YearsInCurrentRole , YearsSinceLastPromotion these are the important variables. For the second component Performance Rating and PerformanceSalaryHike are major contributors. For the third component MonthlyIncome, PerformanceRating, NumCompaniesWorked , YearsINCurrentRole are major contributors.

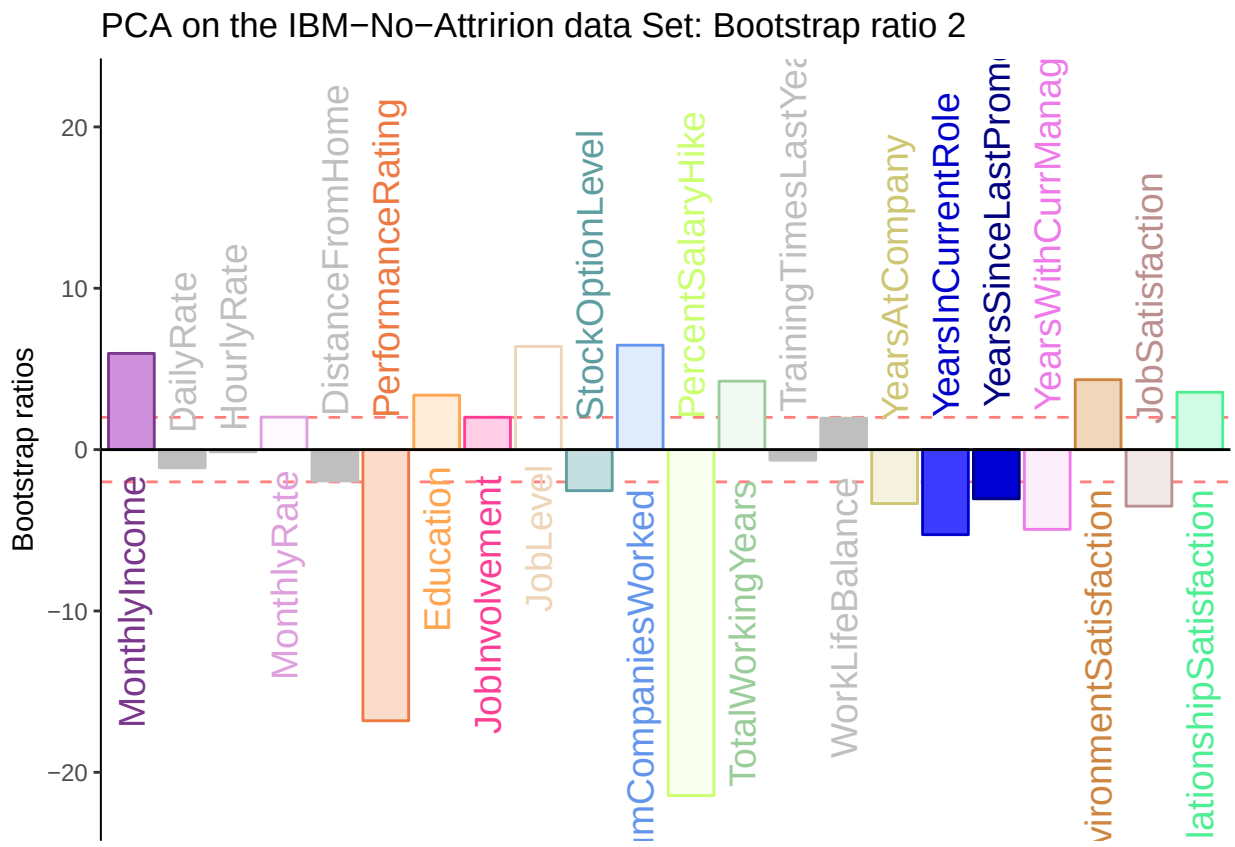
3.5 Bootstrap Ratios

Usually, if Bootstrap Ratio comes out to be greater than 1.96 we say that it's a significant variable or an important contributor on it's own.

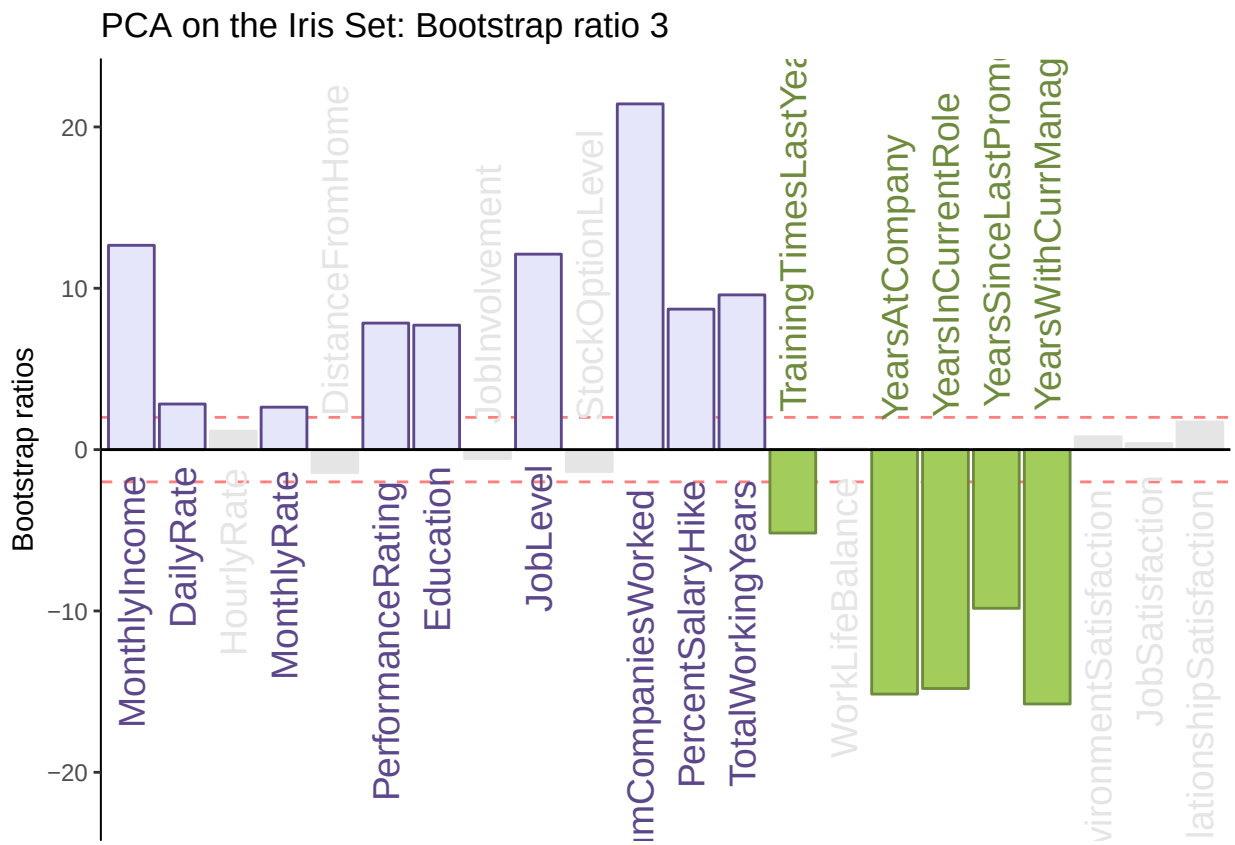
```
BR <- resPCA.inf$Inference.Data$fj.boots$tests$boot.ratios
laDim = 1
ba001.BR1 <- PrettyBarPlot2(BR[,laDim],
                            threshold = 2,
                            font.size = 5,
                            color4bar = gplots::col2hex(col4J.ibm),
                            main = paste0( 'PCA on the IBM-No-Attririon data Set: Bootstrap ratio ',laD
                            ylab = 'Bootstrap ratios'
                            #ylim = c(1.2*min(BR[,laDim]), 1.2*max(BR[,laDim]))
)
print(ba001.BR1)
```




```
#
laDim = 2
ba002.BR2 <- PrettyBarPlot2(BR[,laDim],
                            threshold = 2,
                            font.size = 5,
                            color4bar = gplots::col2hex(col4J.ibm),
                            main = paste0(
                              'PCA on the IBM-No-Attririon data Set: Bootstrap ratio ',laDim),
                            ylab = 'Bootstrap ratios'
)
print(ba002.BR2)
```




```
laDim = 3
ba002.BR3 <- PrettyBarPlot2(BR[,laDim],
                            threshold = 2,
                            font.size = 5,
                            main = paste0(
                              'PCA on the Iris Set: Bootstrap ratio ',laDim),
                            ylab = 'Bootstrap ratios'
)
print(ba002.BR3)
```



Here, we say that Monthly income,Education,Years of experience is negatively correlated with JobInvolvement for the first dimension. For the second dimension , we can say that Monthly income is negatively correlated with the years of experience and Education is also negatively correlated with Years of Experience.

3.6 Summary

When we interpret the factor scores and loadings together, the PCA revealed:

- Component 1:
- Usually the Research and Development people have more work-experience & Monthly Income in comparison to other department type
- Generally, Managers and Directors are the ones with PhD & Master's with higher job level, Sales representative and Lab Technician have no college education while Healthcare Representatives & Manufacturing Directors have Bachelor's degree
- People who are Managers and Research Directors have the highest pay and are more seasoned while people who are Sales representative and Lab Technician have low pay and less experience
- It is also observed that the Mangers and Research Directors travel the most, could be their meetings and conferences while the HR, Sales representative,Lab Technician, Research Scientists travel occasionally
- Component 2: Human Resources people have high Performance Rating

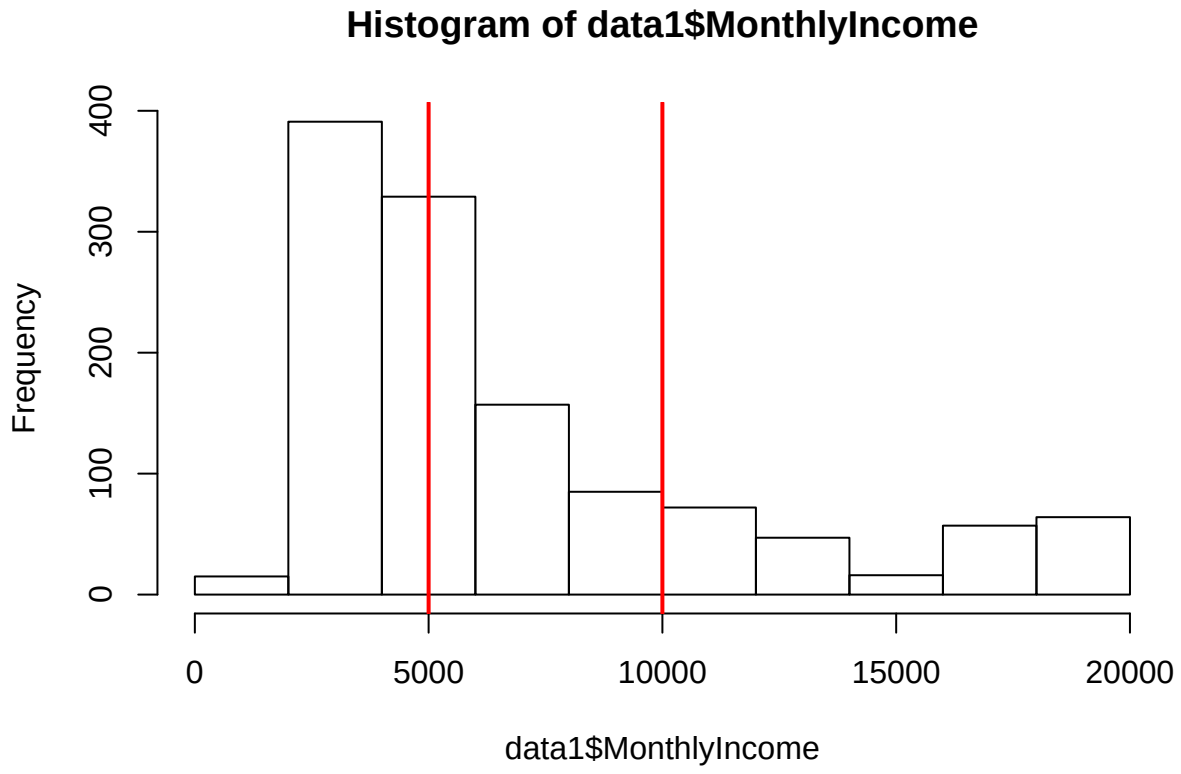
4 Multiple Correspondance Analysis

Multiple correspondence analysis (MCA) is an extension of correspondence analysis (CA) which allows one to analyze the pattern of relationships of several categorical dependent variables. MCA is used to analyze a set of observations described by a set of nominal variables. Each nominal variable comprises several levels, and each of these levels is coded as a binary variable. MCA can also accommodate quantitative variables by recoding them as bins.Because MCA has been (re)discovered many times, equivalent methods are known under several different names such as optimal scaling, optimal or appropriate scoring, dual scaling, homogeneity analysis, scalogram analysis, and quantification method. Technically MCA is obtained by using a standard correspondence analysis on an indicator matrix (a matrix whose entries are 0 or 1). The percentages of explained variance need to be corrected, and the correspondence analysis interpretation of interpoint distances needs to be adapted. Also, MCA is helpful in capturing the non-linear relationship of the data.

4.1 Binning the quantitative data

The goal of binning the quantitative data is to convert them into nominal scale so that presence of non-linear realtionship in data can be detected. Also, binning is usually done by diving the variable into equal number of bins of tables or the correlation between the divided bins of tables should be greater than 0.80

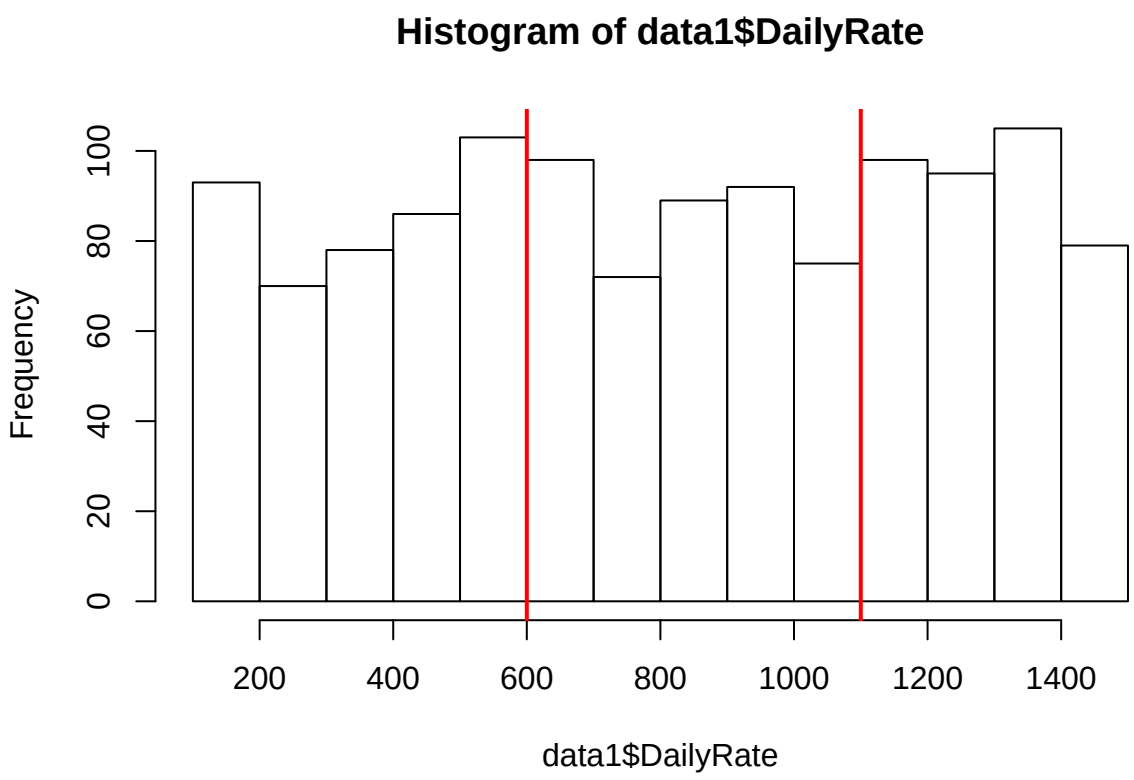
```
hist(data1$MonthlyIncome)
abline(v = c(5000,10000), col = "red", lwd =2)
```




```
data1$MonthlyIncome <- cut(data1$MonthlyIncome,breaks = c(min(data1$MonthlyIncome)-1,5000 ,10000, max(da
cor(as.numeric(data1$MonthlyIncome),my_data$MonthlyIncome,method = "spearman")
```

```
## [1] 0.9229362
```

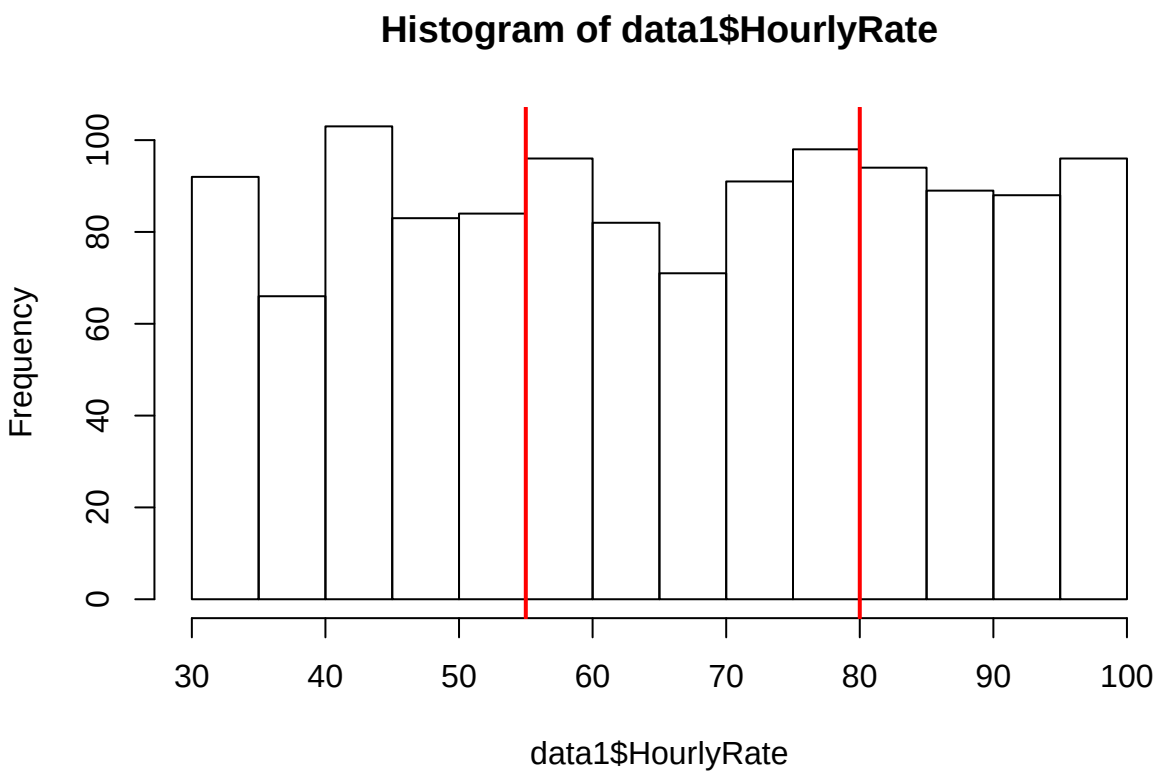
```
hist(data1$DailyRate)
abline(v = c(600,1100), col = "red", lwd =2)
```




```
data1$DailyRate <- cut(data1$DailyRate,breaks = c(min(data1$DailyRate)-1,600 ,1100, max(data1$DailyRate)
cor(as.numeric(data1$DailyRate),my_data$DailyRate,method = "spearman")
```

```
## [1] 0.9422108
```

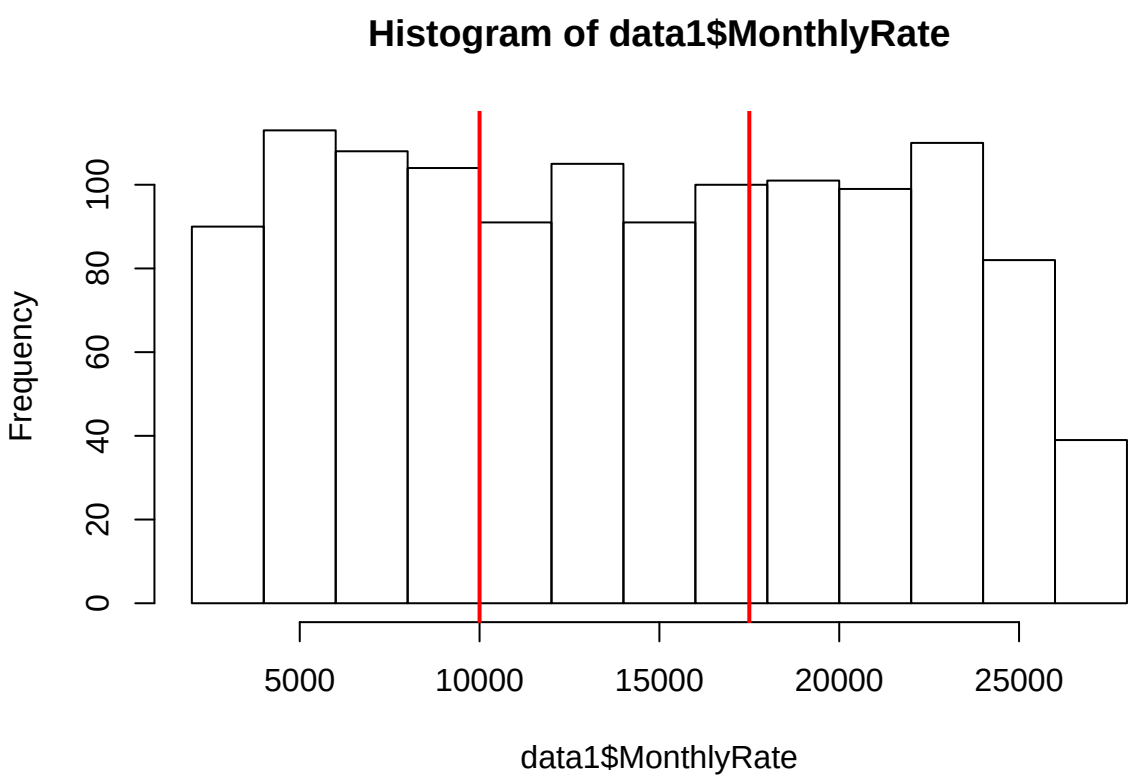
```
hist(data1$HourlyRate)
abline(v = c(55,80), col = "red", lwd =2)
```




```
data1$HourlyRate <- cut(data1$HourlyRate,breaks = c(min(data1$HourlyRate)-1,55 ,80, max(data1$HourlyRate
cor(as.numeric(data1$HourlyRate),my_data$HourlyRate,method = "spearman")
```

```
## [1] 0.9419043
```

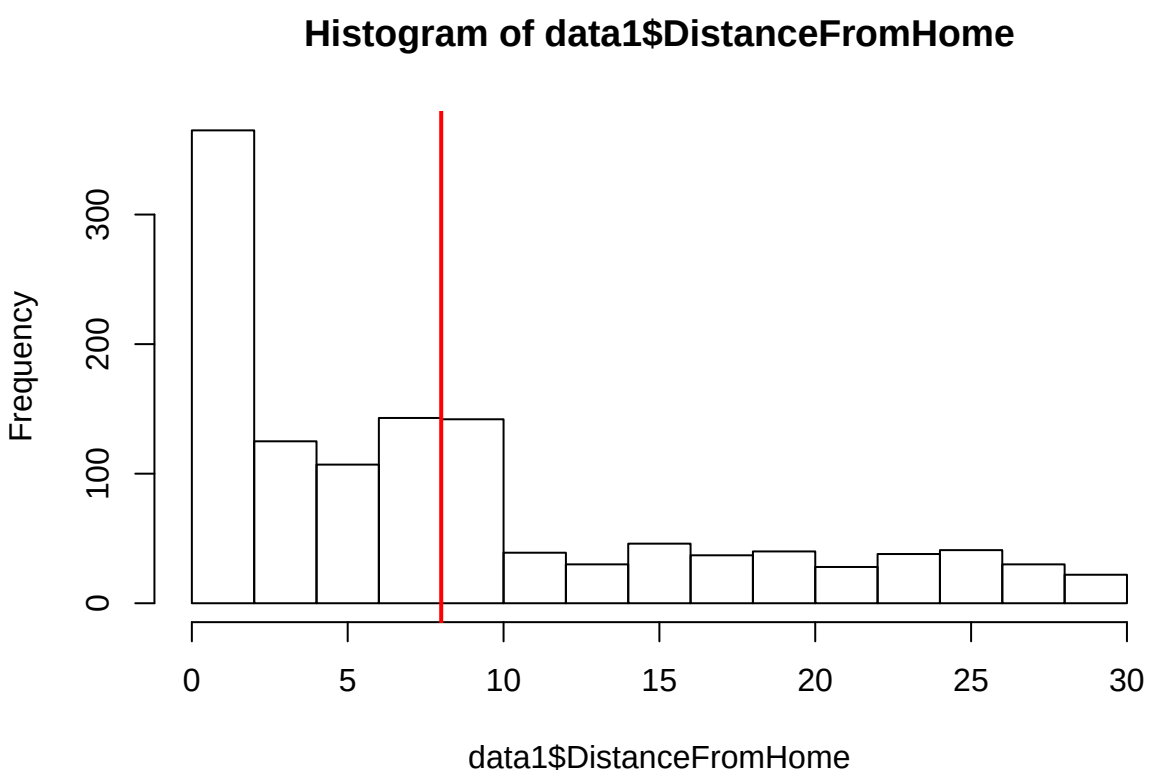
```
hist(data1$MonthlyRate)
abline(v = c(10000,17500), col = "red", lwd =2)
```




```
data1$MonthlyRate <- cut(data1$MonthlyRate,breaks = c(min(data1$MonthlyRate)-1,10000 ,17500, max(data1$
cor(as.numeric(data1$MonthlyRate),my_data$MonthlyRate,method = "spearman")
```

```
## [1] 0.9412661
```

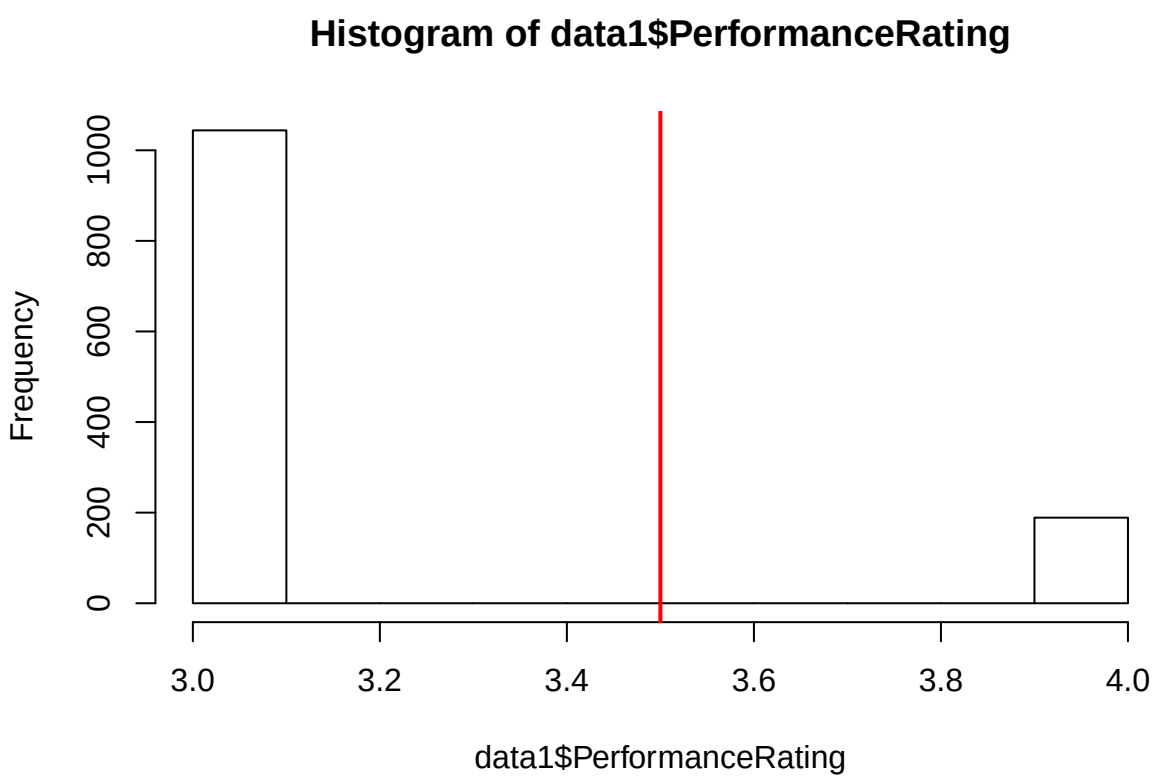
```
hist(data1$DistanceFromHome)
abline(v = c(8), col = "red", lwd =2)
```



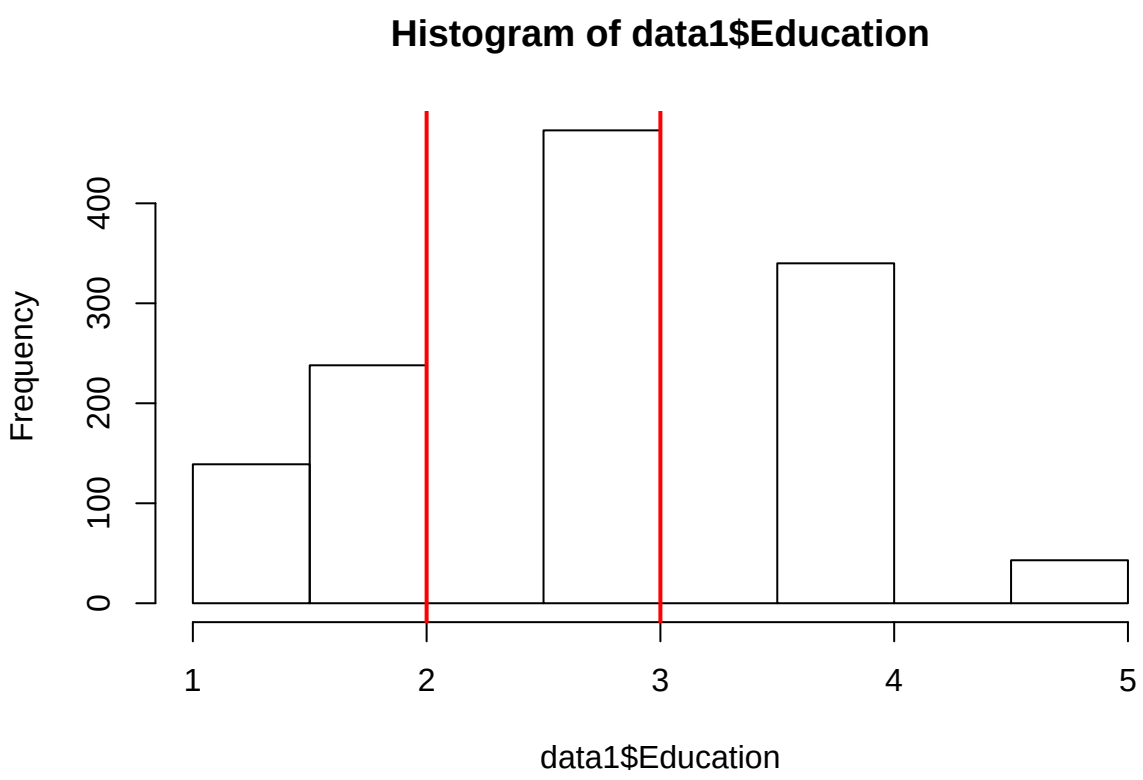

```
data1$DistanceFromHome <- cut(data1$DistanceFromHome,breaks = c(min(data1$DistanceFromHome)-1,6 , max(da
cor(as.numeric(data1$DistanceFromHome),my_data$DistanceFromHome,method = "spearman")
```

```
## [1] 0.8689751
```

```
hist(data1$PerformanceRating)
abline(v = c(3.5), col = "red", lwd =2)
```



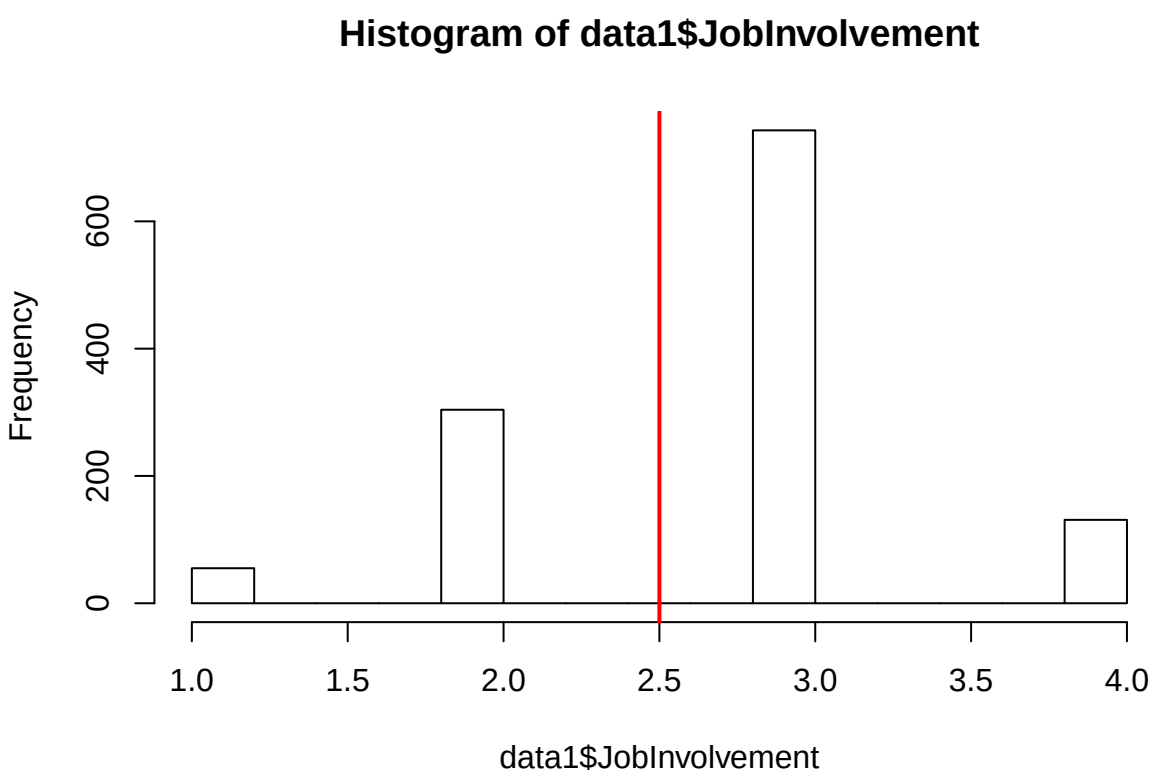

```r
hist(data1$Education)
abline(v = c(2,3), col = "red", lwd =2)
```




```
data1$Education <- cut(data1$Education,breaks = c(min(data1$Education)-1, 2,3 , max(data1$Education)+1)
cor(as.numeric(data1$Education),my_data$Education,method = "spearman")
```

```
## [1] 0.9840498
```

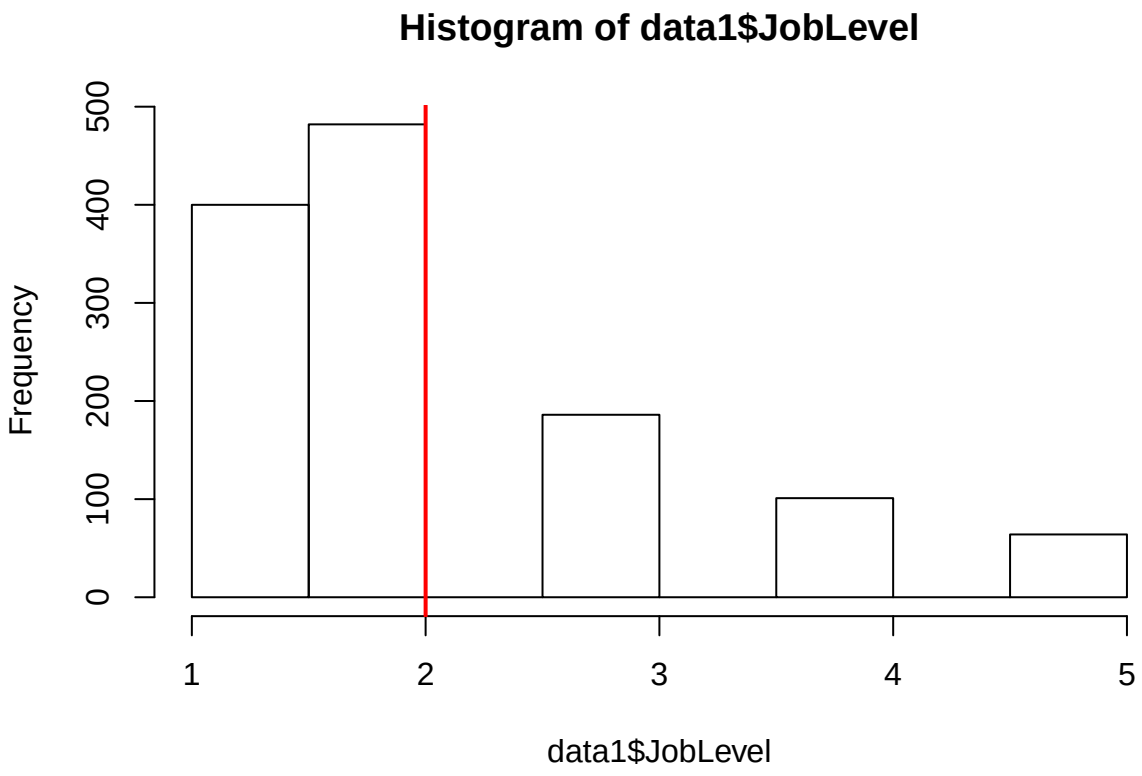
```
hist(data1$JobInvolvement)
abline(v = c(2.5), col = "red", lwd =2)
```




```
data1$JobInvolvement <- cut(data1$JobInvolvement,breaks = c(min(data1$JobInvolvement)-1,2.5, max(data1$
cor(as.numeric(data1$JobInvolvement),my_data$JobInvolvement,method = "spearman")
```

```
## [1] 0.8996954
```

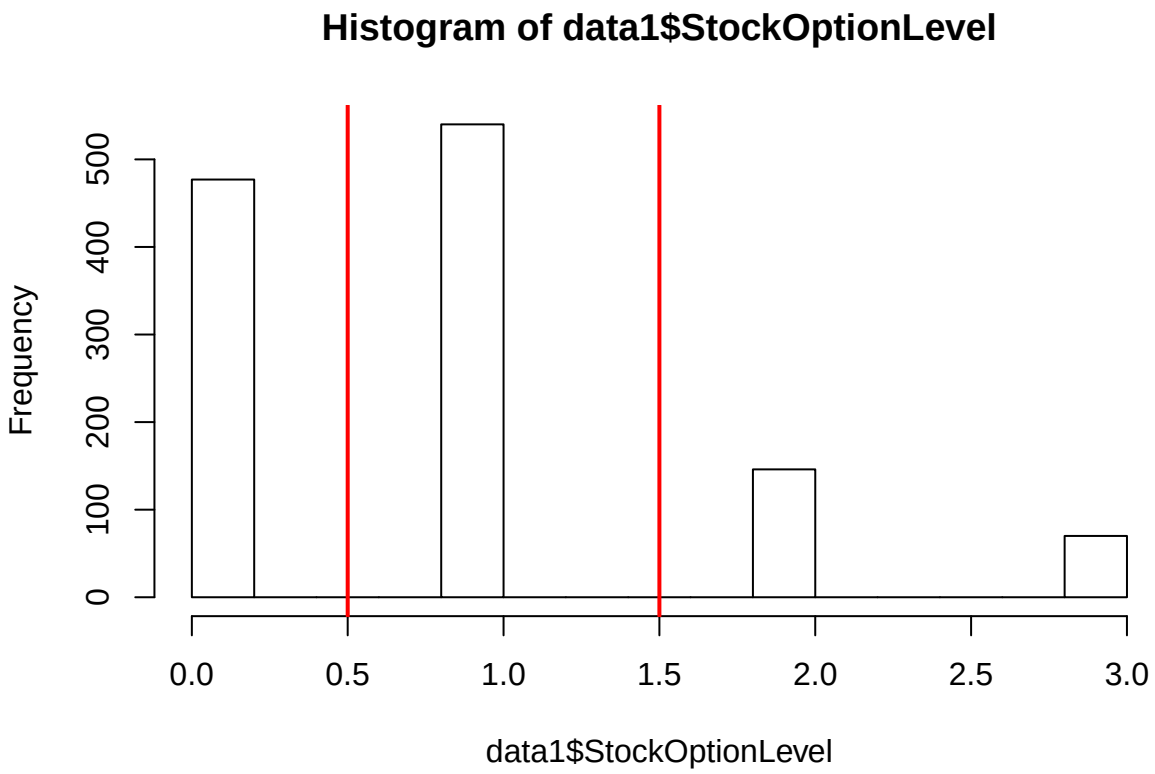
```
hist(data1$JobLevel)
abline(v = c(2), col = "red", lwd =2)
```




```
data1$JobLevel <- cut(data1$JobLevel,breaks = c(min(data1$JobLevel)-1,2, max(data1$JobLevel)+1),labels =
cor(as.numeric(data1$JobLevel),my_data$JobLevel,method = "spearman")
```

```
## [1] 0.8229676
```

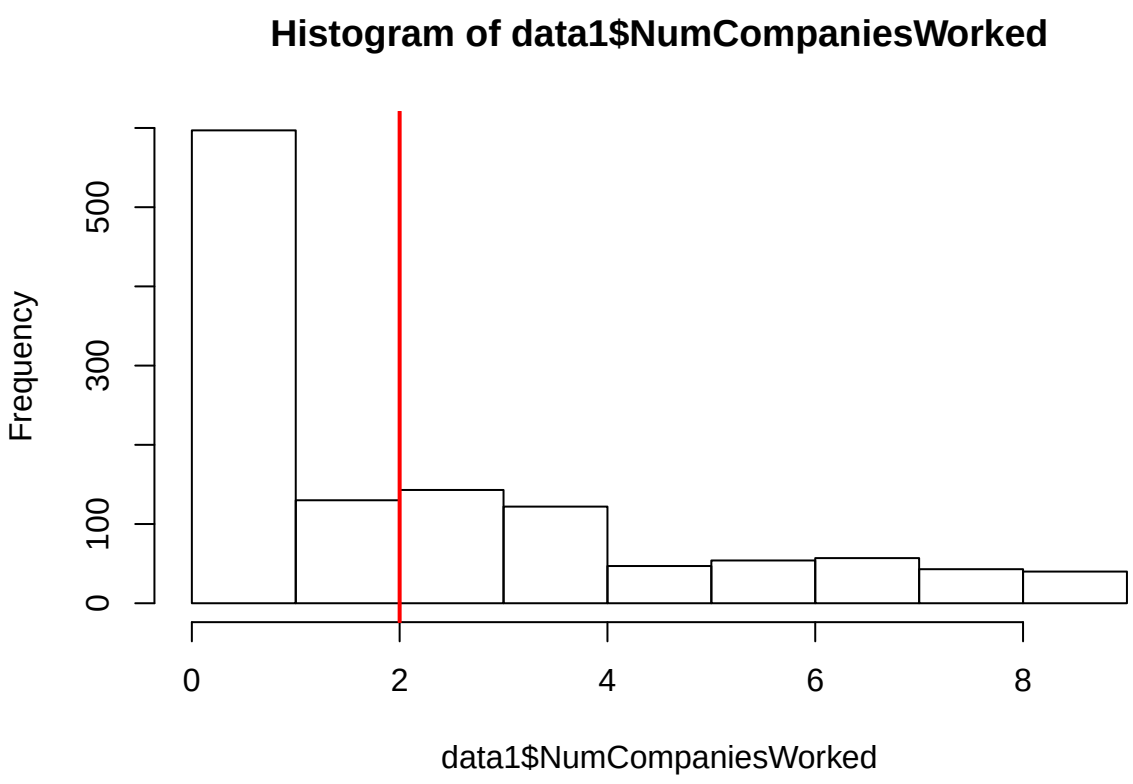
```
hist(data1$StockOptionLevel)
abline(v = c(0.5,1.5), col = "red", lwd =2)
```




```
data1$StockOptionLevel <- cut(data1$StockOptionLevel,breaks=c(min(data1$StockOptionLevel)-1,0.5,1.5,max
cor(as.numeric(data1$StockOptionLevel),my_data$StockOptionLevel,method = "spearman")
```

```
## [1] 0.9979348
```

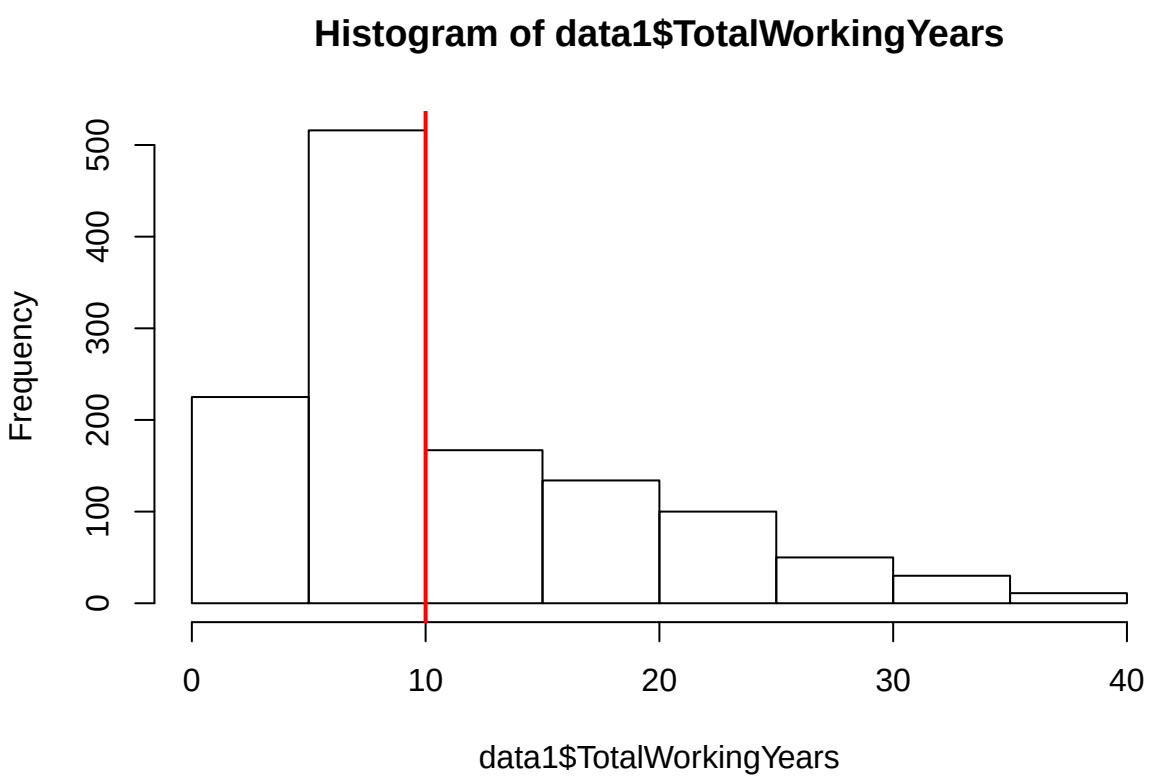
```
hist(data1$NumCompaniesWorked)
abline(v = c(2), col = "red", lwd =2)
```




```
data1$NumCompaniesWorked <- cut(data1$NumCompaniesWorked,breaks = c(min(data1$NumCompaniesWorked)-1, 2,
cor(as.numeric(data1$NumCompaniesWorked),my_data$NumCompaniesWorked,method = "spearman")
```

```
## [1] 0.8728502
```

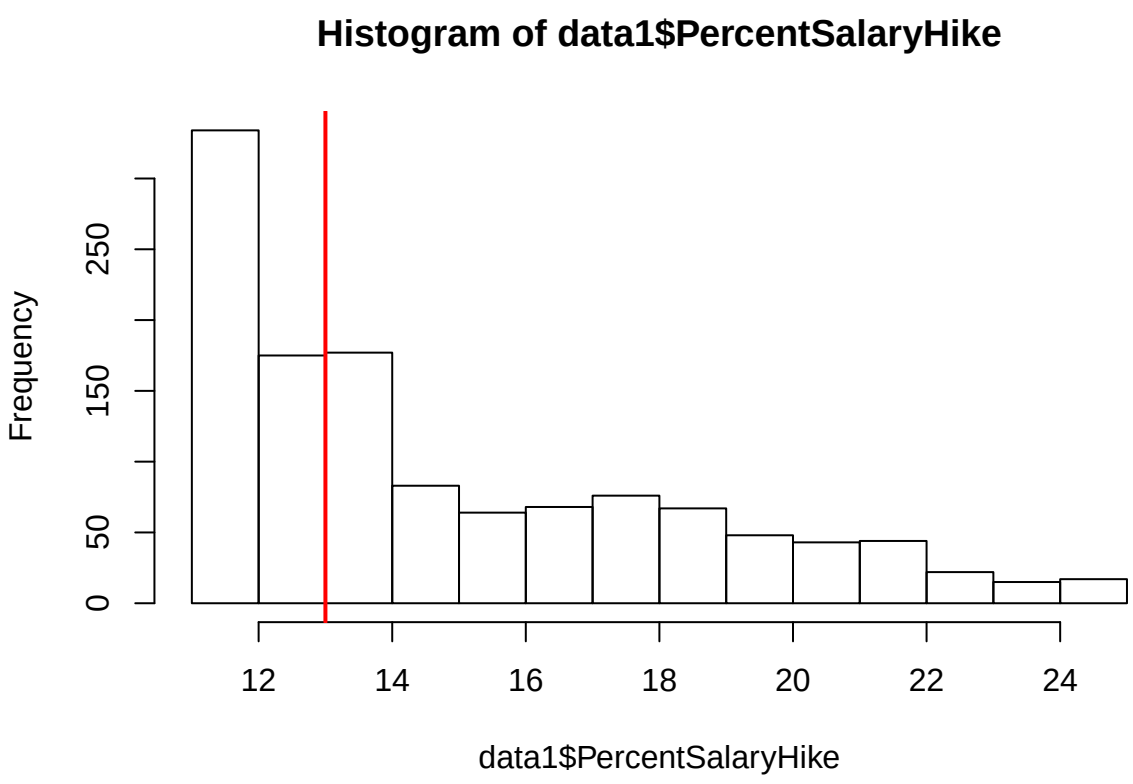
```
hist(data1$TotalWorkingYears)
abline(v = c(10), col = "red", lwd =2)
```




```
data1$TotalWorkingYears <- cut(data1$TotalWorkingYears,breaks = c(min(data1$TotalWorkingYears)-1, 10, ma
cor(as.numeric(data1$TotalWorkingYears),my_data$TotalWorkingYears,method = "spearman")
```

```
## [1] 0.8502664
```

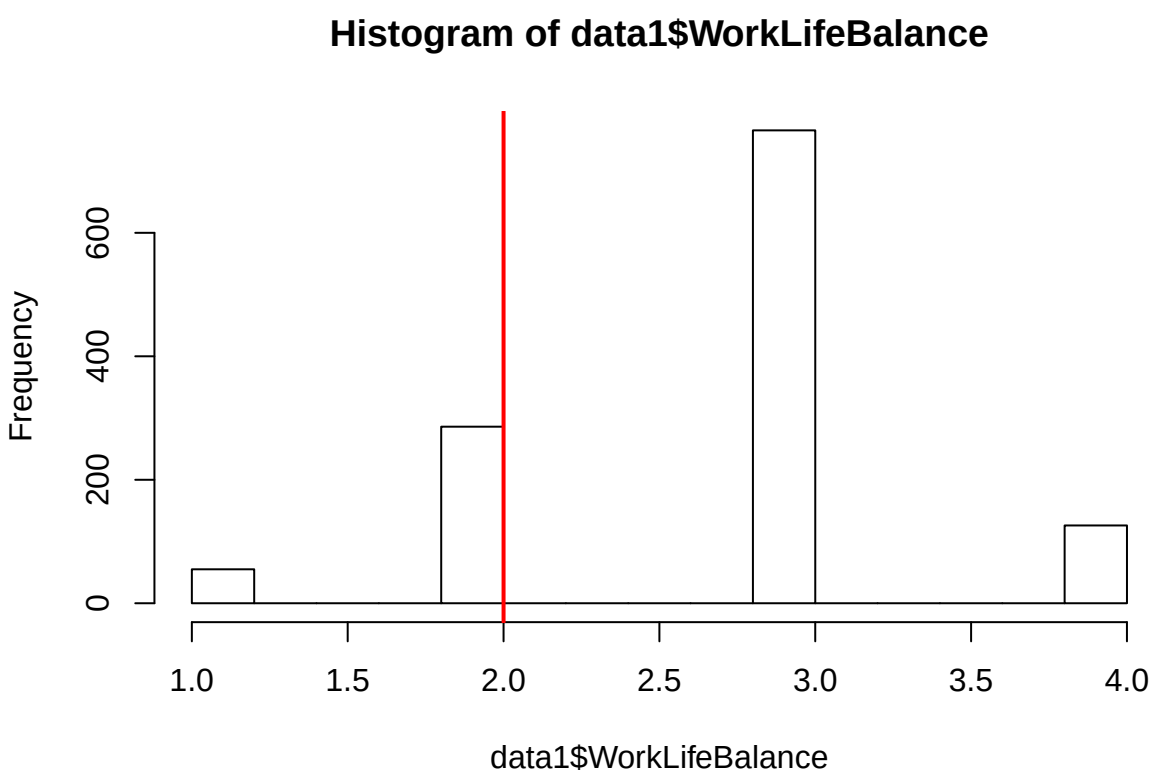
```
hist(data1$PercentSalaryHike)
abline(v = c(13), col = "red", lwd =2)
```




```
data1$PercentSalaryHike <- cut(data1$PercentSalaryHike,breaks = c(min(data1$PercentSalaryHike)-1, 13, ma
cor(as.numeric(data1$PercentSalaryHike),my_data$PercentSalaryHike,method = "spearman")
```

```
## [1] 0.8579005
```

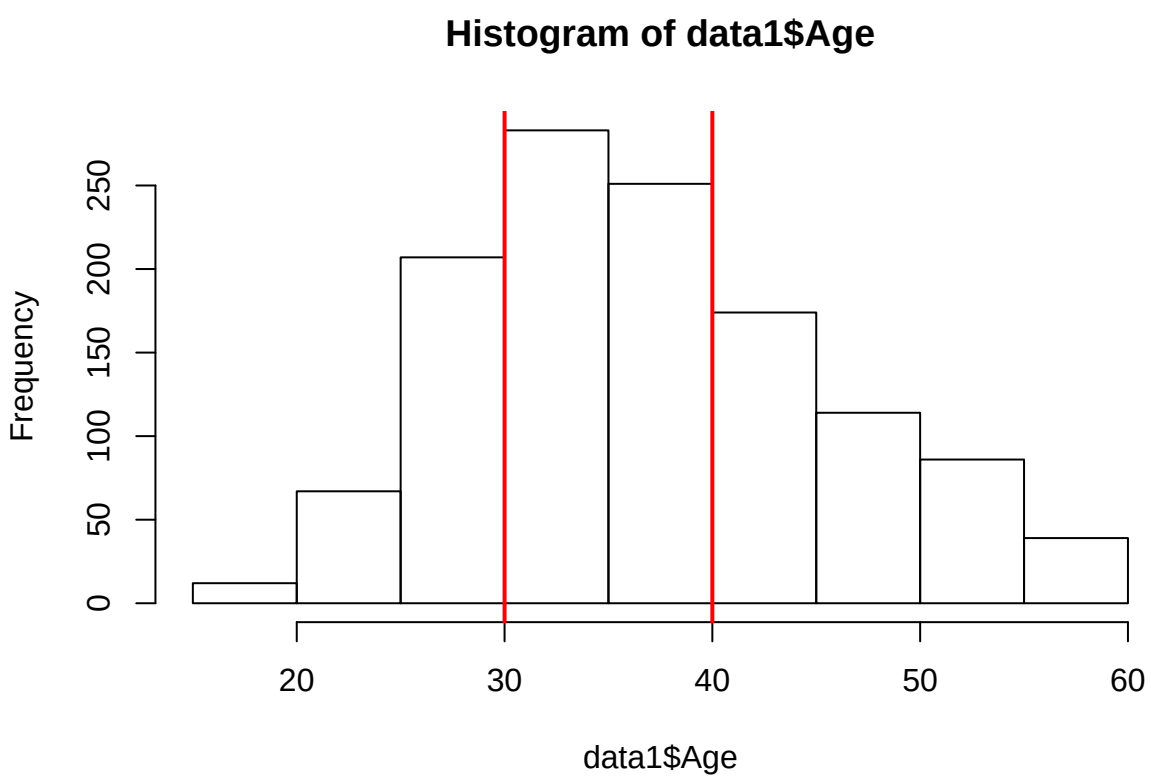
```
hist(data1$WorkLifeBalance)
abline(v = c(2), col = "red", lwd =2)
```




```
data1$WorkLifeBalance <- cut(data1$WorkLifeBalance,breaks = c(min(data1$WorkLifeBalance)-1, 2, max(data
cor(as.numeric(data1$PercentSalaryHike),my_data$PercentSalaryHike,method = "spearman")
```

```
## [1] 0.8579005
```

```
hist(data1$Age)
abline(v = c(30,40), col = "red", lwd =2)
```




```
data1$Age <- cut(data1$Age,breaks = c(min(data1$Age)-1, 30,40, max(data1$Age)+1),labels = c(1,2,3))
cor(as.numeric(data1$Age),my_data$Age,method = "spearman")
```

```
## [1] 0.9326494
```

```
hist(data1$TrainingTimesLastYear)
abline(v = c(2), col = "red", lwd =2)
```


```
data1$TrainingTimesLastYear <- cut(data1$TrainingTimesLastYear,breaks = c(min(data1$TrainingTimesLastYea
cor(as.numeric(data1$TrainingTimesLastYear),my_data$TrainingTimesLastYear,method = "spearman")
```

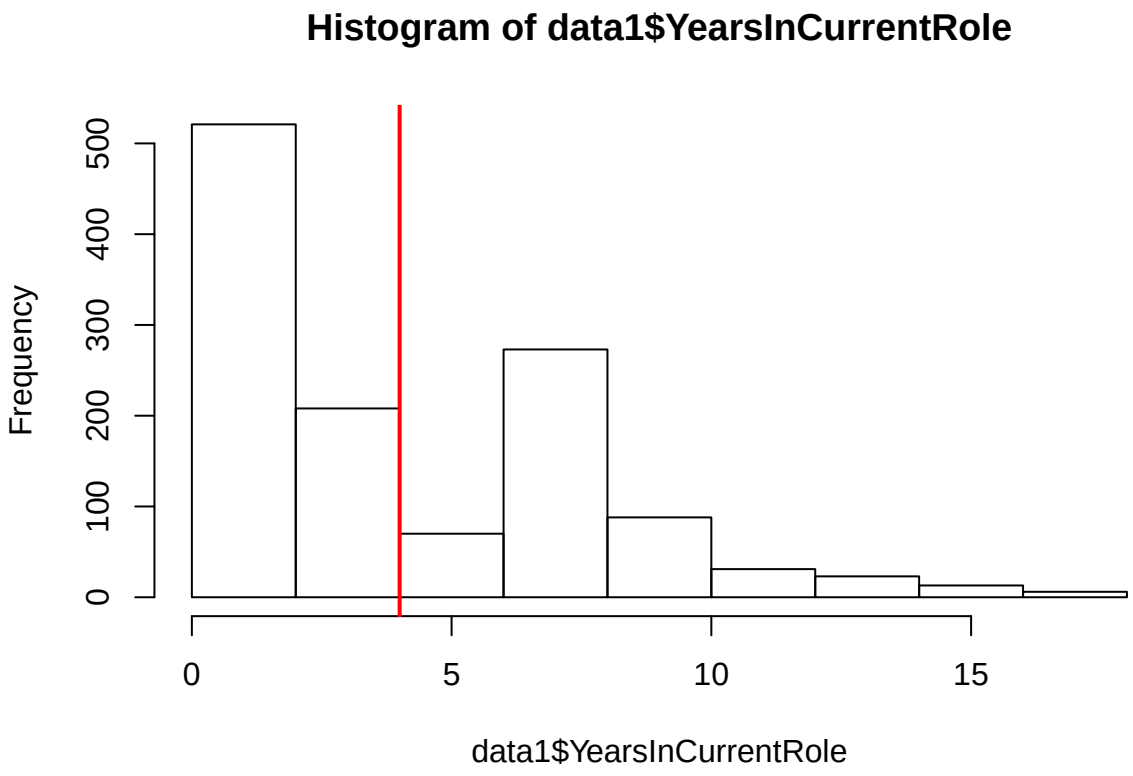
```
## [1] 0.9024219
```

```
hist(data1$YearsAtCompany)
abline(v = c(5), col = "red", lwd =2)
```


```
data1$YearsAtCompany <- cut(data1$YearsAtCompany,breaks = c(min(data1$YearsAtCompany)-1, 5, max(data1$Y
cor(as.numeric(data1$YearsAtCompany),my_data$YearsAtCompany,method = "spearman")
```

```
## [1] 0.8689195
```

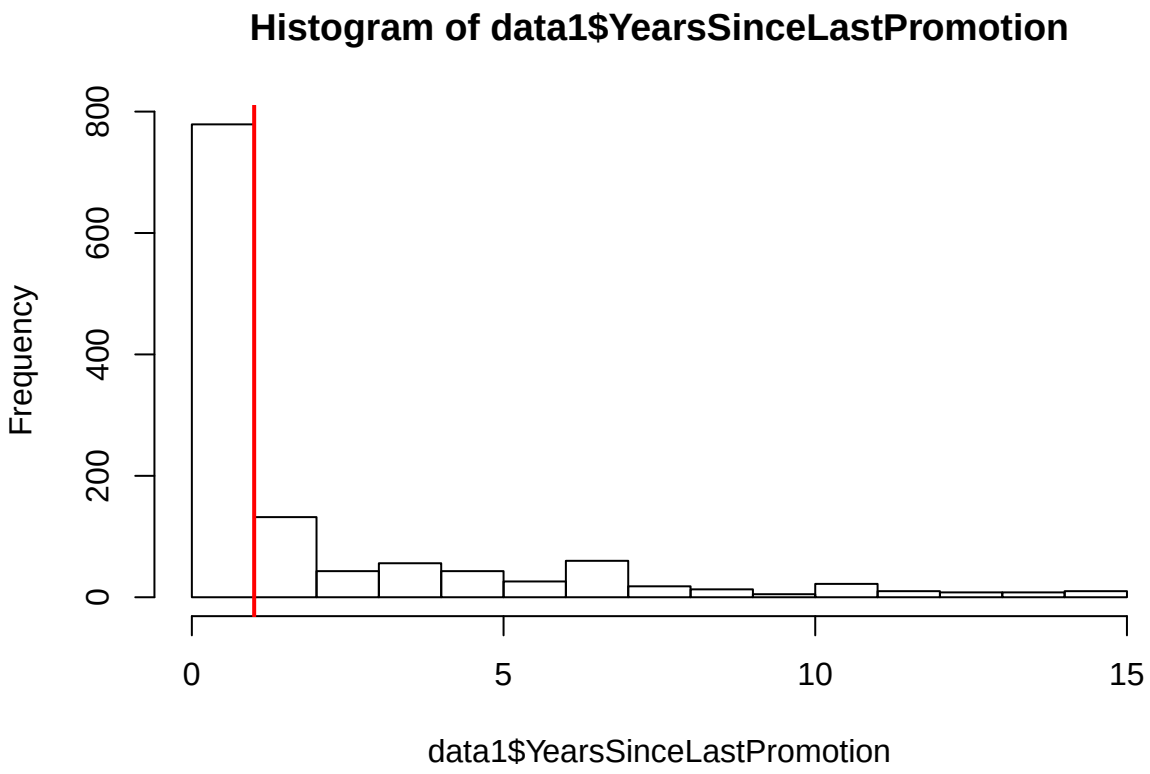
```
hist(data1$YearsInCurrentRole)
abline(v = c(4), col = "red", lwd =2)
```




```
data1$YearsInCurrentRole <- cut(data1$YearsInCurrentRole,breaks = c(min(data1$YearsInCurrentRole)-1, 4,
cor(as.numeric(data1$YearsInCurrentRole),my_data$YearsInCurrentRole,method = "spearman")
```

```
## [1] 0.8615273
```

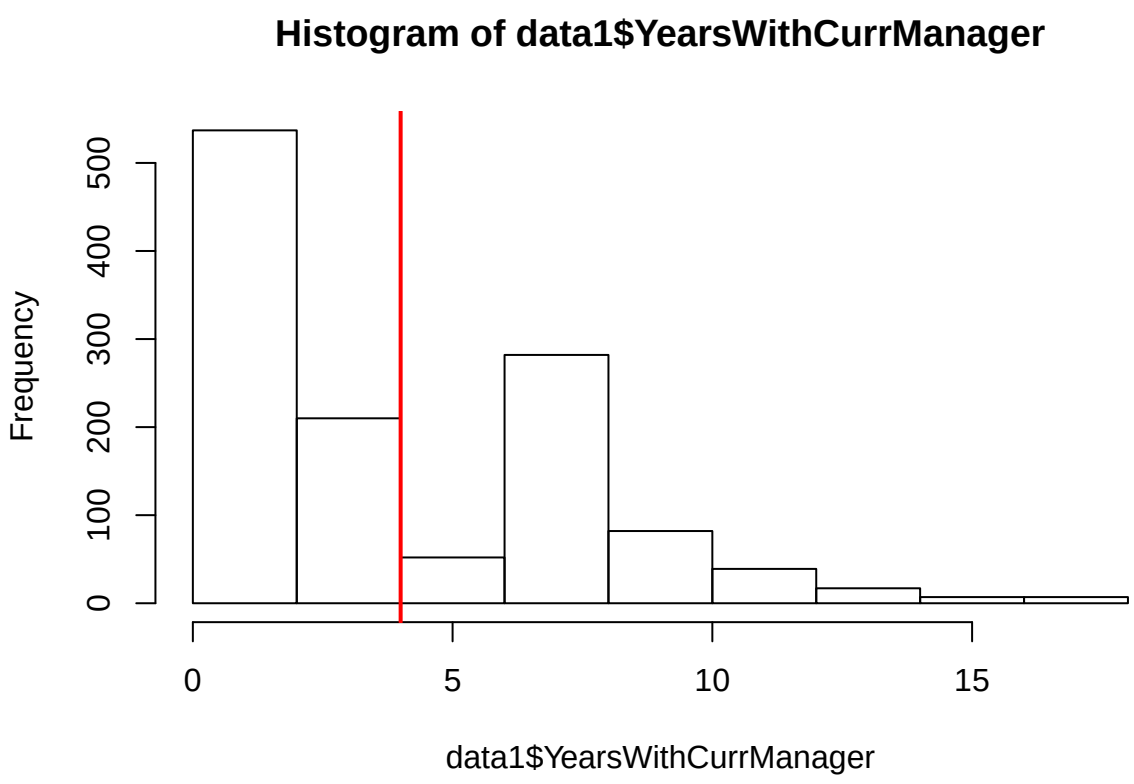
```
hist(data1$YearsSinceLastPromotion)
abline(v = c(1), col = "red", lwd =2)
```




```
data1$YearsSinceLastPromotion <- cut(data1$YearsSinceLastPromotion,breaks = c(min(data1$YearsSinceLastPr
cor(as.numeric(data1$YearsSinceLastPromotion),my_data$YearsSinceLastPromotion,method = "spearman")
```

```
## [1] 0.8676085
```

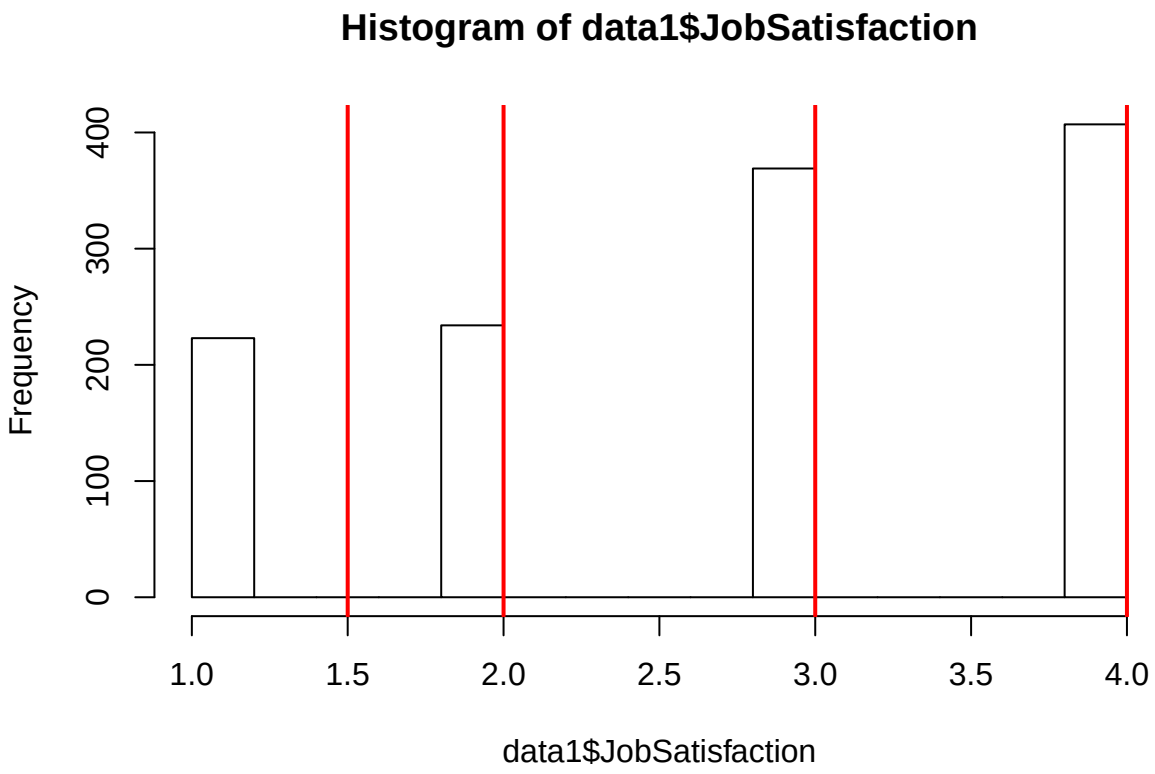
```
hist(data1$YearsWithCurrManager)
abline(v = c(4), col = "red", lwd =2)
```



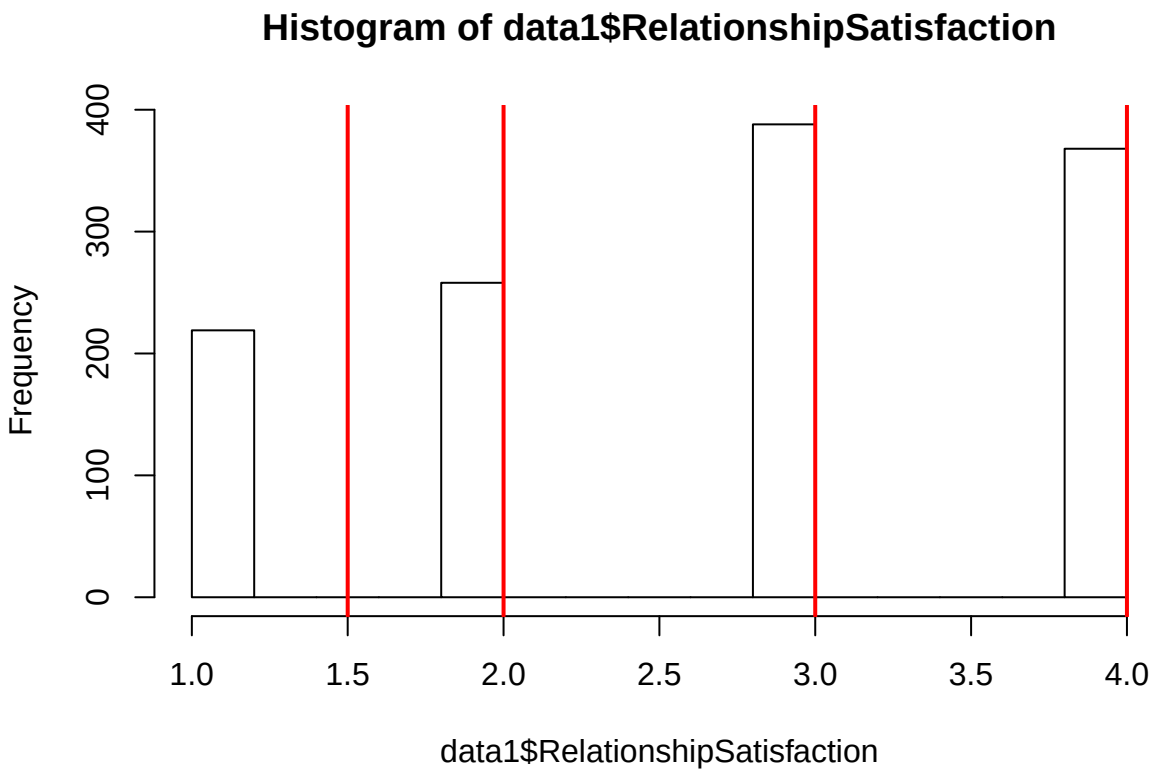

```
data1$YearsWithCurrManager <- cut(data1$YearsWithCurrManager,breaks = c(min(data1$YearsWithCurrManager)-
cor(as.numeric(data1$YearsWithCurrManager),my_data$YearsWithCurrManager,method = "spearman")
```

```
## [1] 0.855894
```

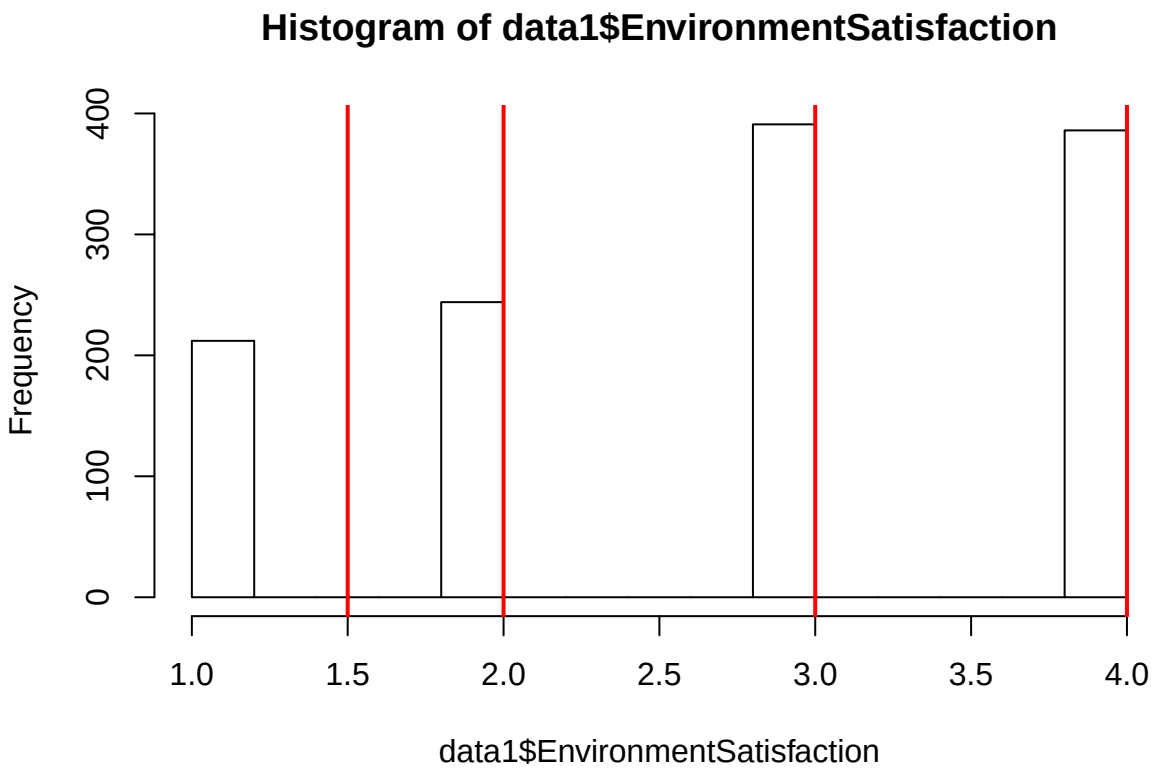
```
hist(data1$JobSatisfaction)
abline(v = c(1.5,2,3,4), col = "red", lwd =2)
```




```
hist(data1$RelationshipSatisfaction)
abline(v = c(1.5,2,3,4), col = "red", lwd =2)
```

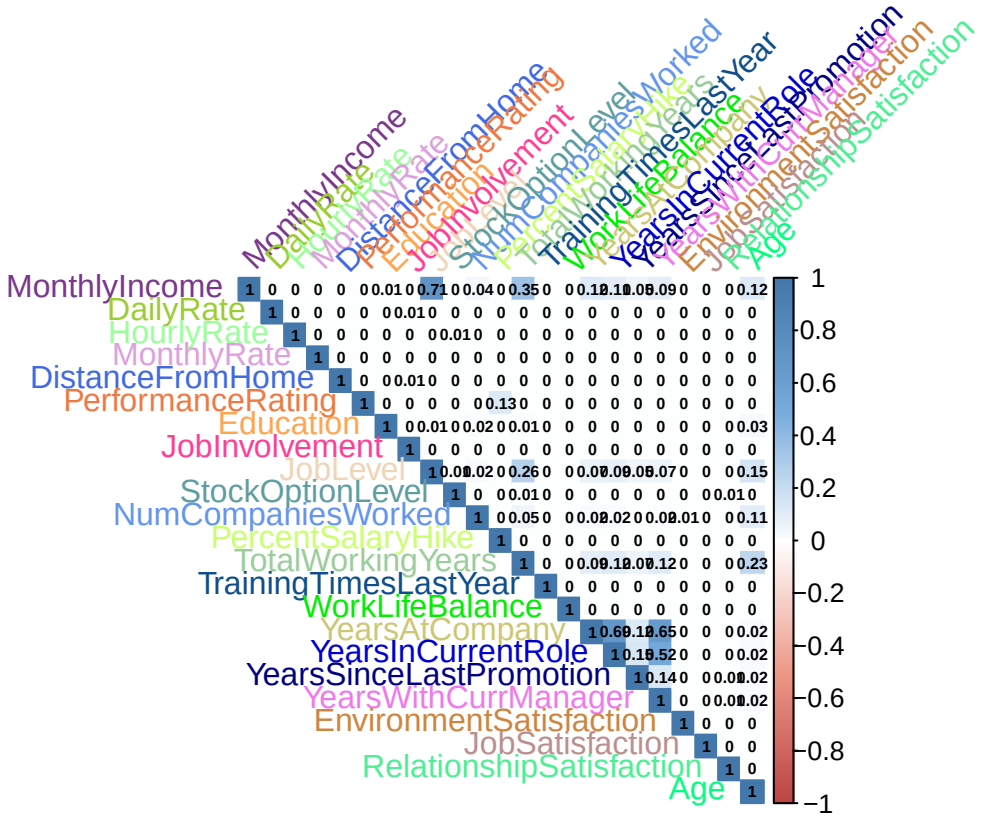



```r
hist(data1$EnvironmentSatisfaction)
abline(v = c(1.5,2,3,4), col = "red", lwd =2)
```



HeatMap

```
color4Var1 <- prettyGraphs::prettyGraphsColorSelection(ncol(data1))
col <- colorRampPalette(c("#BB4444", "#EE9988", "#FFFFFF", "#77AADD", "#4477AA"))

corrMatBurt.list1 <- phi2Mat4BurtTable(data1)
corr4MCA.r <- corrplot::corrplot( as.matrix(corrMatBurt.list1$phi2.mat), method="color", col=col(200),
addCoef.col = "black", tl.col=color4Var1 ,
tl.srt = 45, #Text label color and rotation
number.cex = .5,
diag = TRUE  )
```



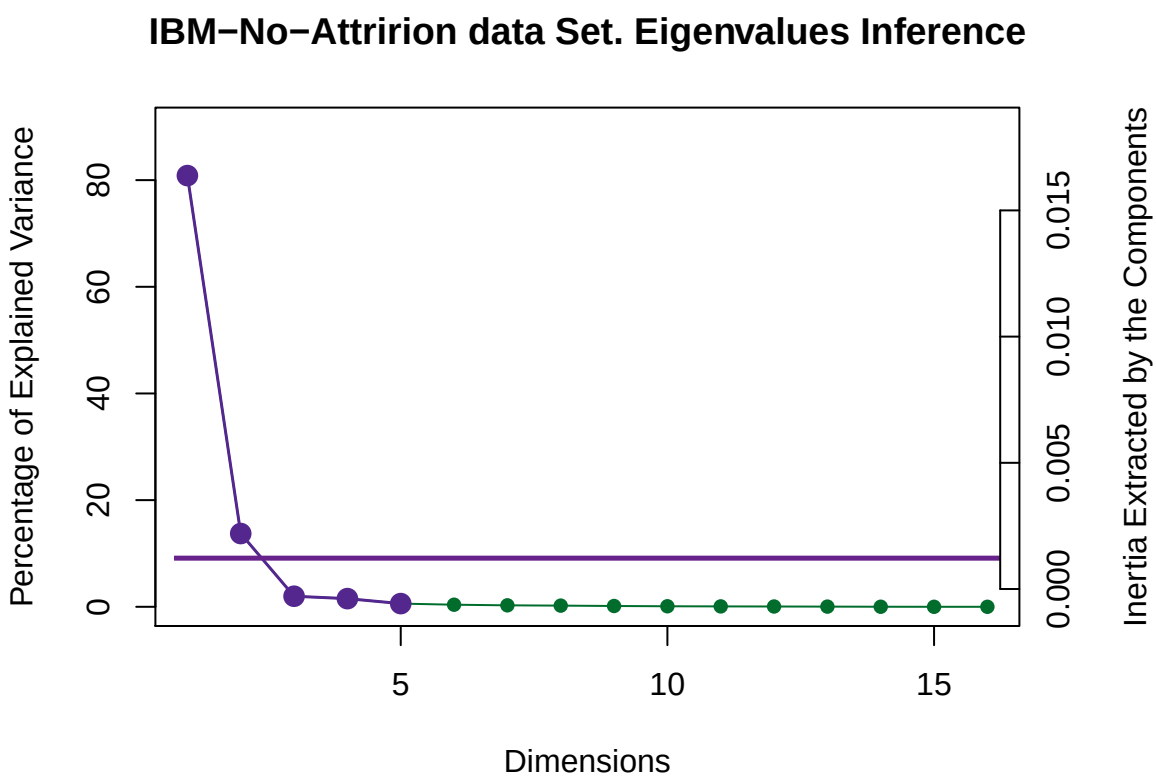
From the heat Map we can see that high correlation is present between the Job Level and Monthly Income, Montly Income/Job Level and Total number of working years,Job level/Total Working years and Age

Scree plot

```
resMCA <- epMCA(data1 ,make_data_nominal = TRUE ,DESIGN = my_data$Department ,make_design_nominal = TRU

resMCA.inf <- epMCA.inference.battery(data1, make_data_nominal = TRUE, DESIGN = my_data$Department ,mak
```

```
resMCA1 <- epMCA(data1 ,make_data_nominal = TRUE ,DESIGN = my_data$Gender ,make_design_nominal = TRUE,g

resMCA.inf1 <- epMCA.inference.battery(data1, make_data_nominal = TRUE, DESIGN = my_data$Gender ,make_d
```

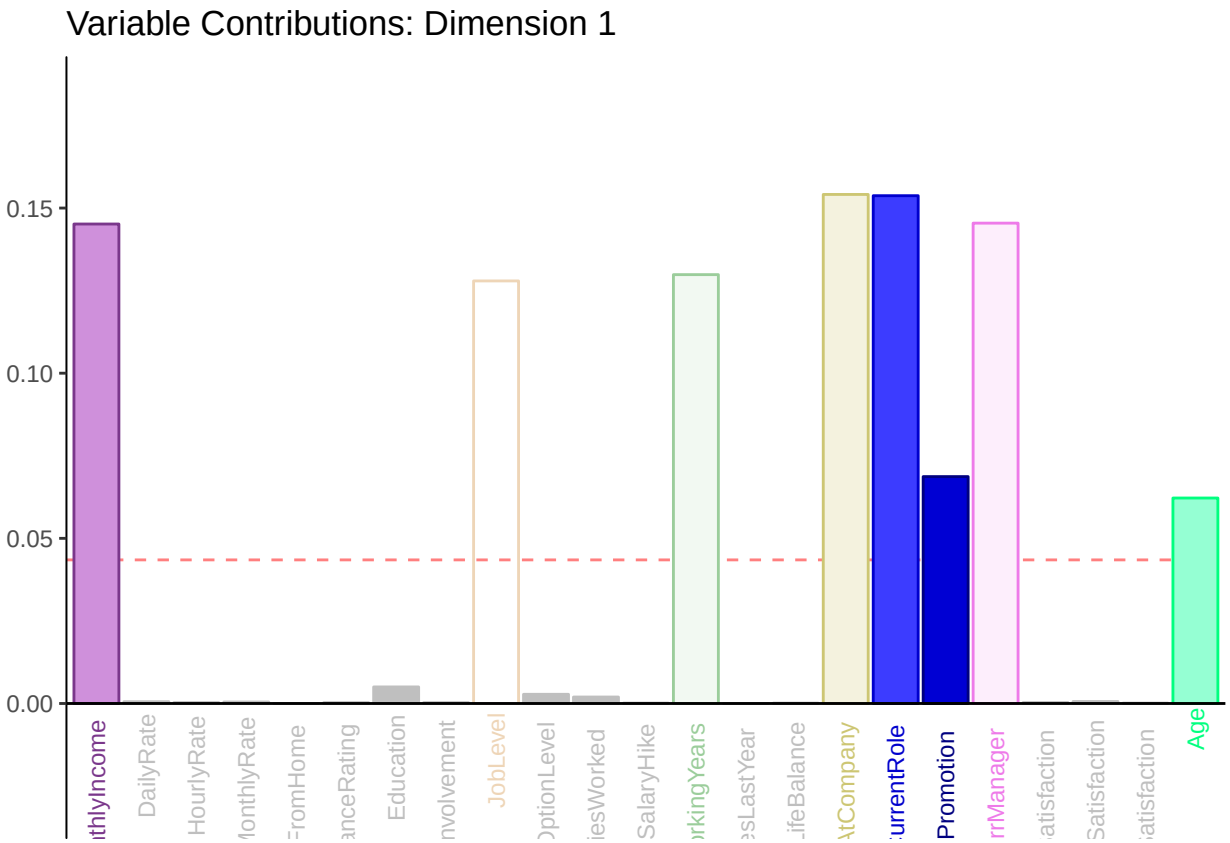
```
PlotScree(ev = resMCA$ExPosition.Data$eigs,
          p.ev =  resMCA.inf$Inference.Data$components$p.vals,
          title = 'IBM-No-Attririon data Set. Eigenvalues Inference',
          plotKaiser = TRUE
)
```



We can say that the Top 2 Components are the important components

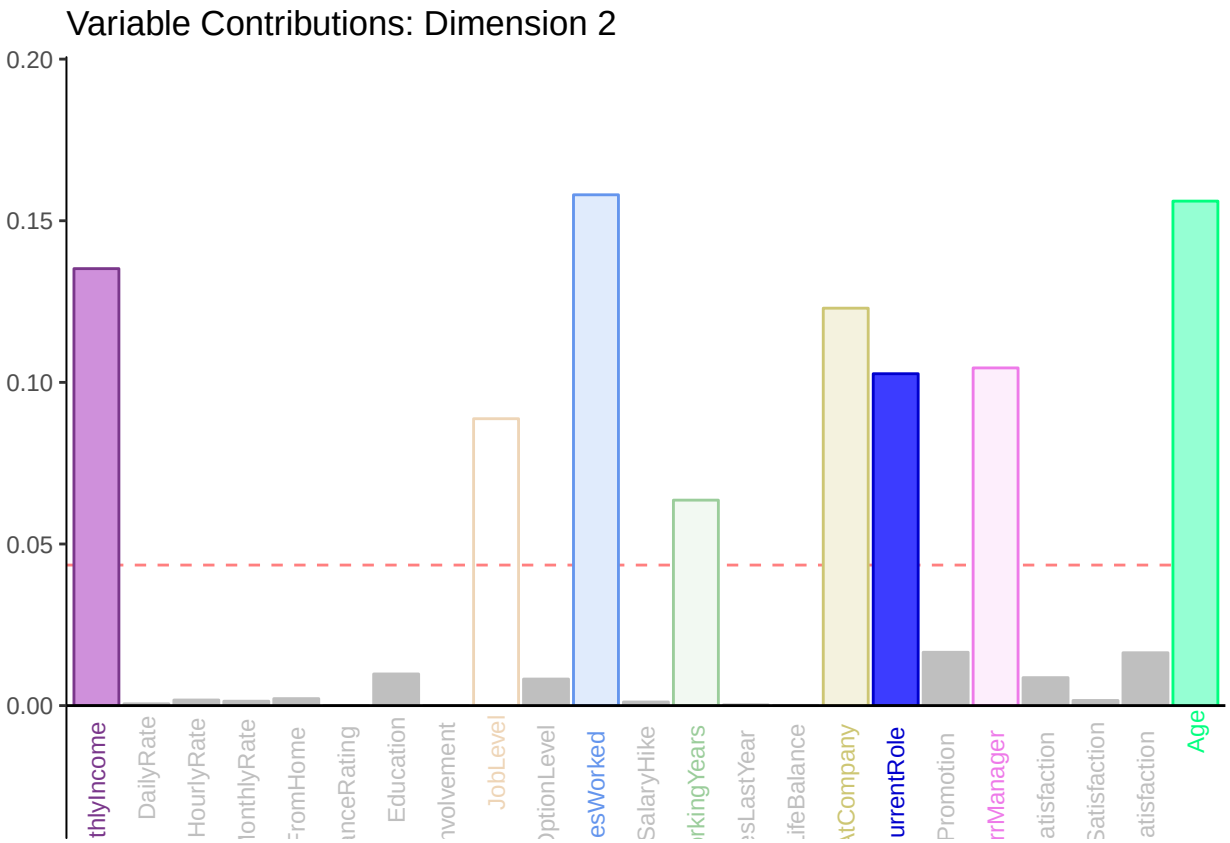
4.2 Variable contributions

```
color <- prettyGraphs::prettyGraphsColorSelection(ncol(data1))
cJ <- resMCA$ExPosition.Data$cj
varCtr <- data4PCCAR::ctr4Variables(cJ)
rownames(color) <- rownames(varCtr)
varCtr1 <- varCtr[,1]
names(varCtr1) <- rownames(varCtr)
a0005.Var.ctr1 <- PrettyBarPlot2(varCtr1, main = 'Variable Contributions: Dimension 1', ylim = c(-.03,1
print(a0005.Var.ctr1)
```



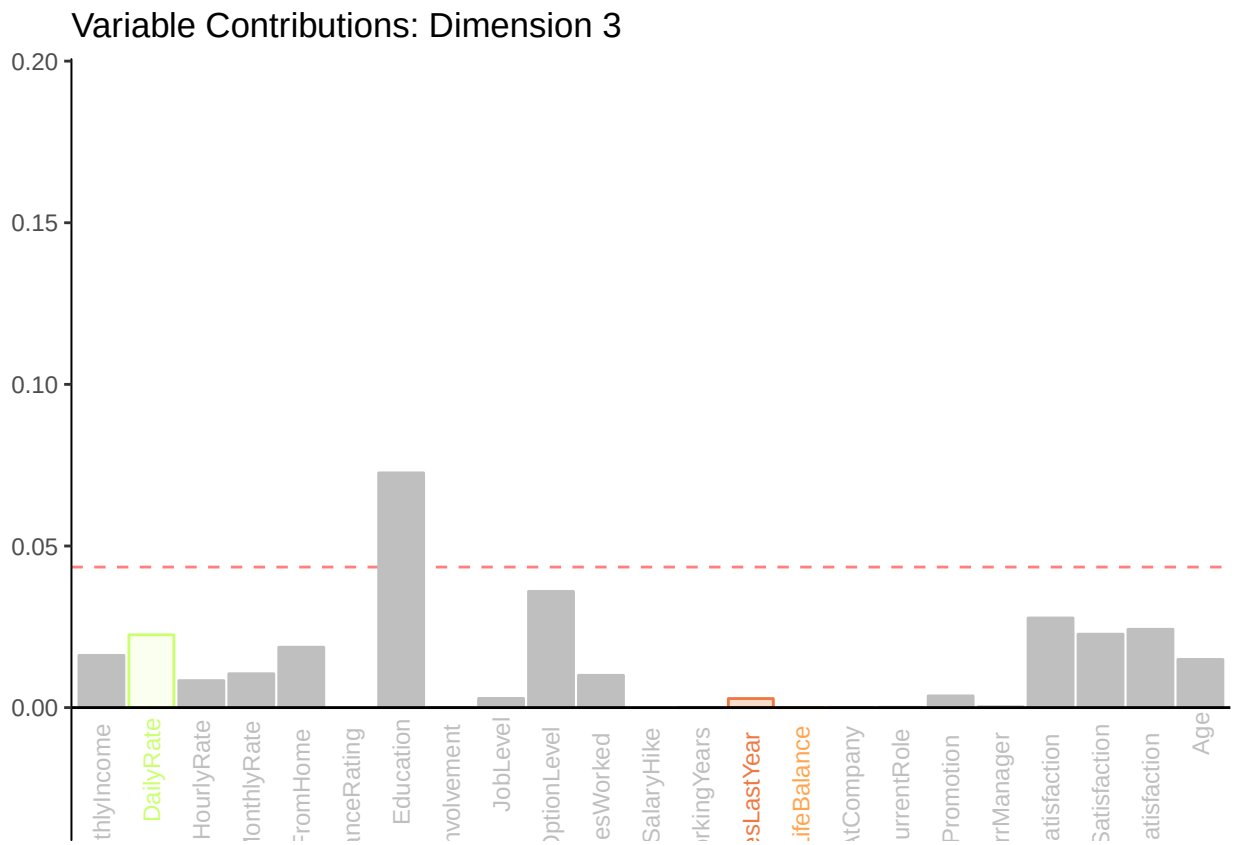
Montly Income, Job Level, Job Role, Age and Years of Experience are again the important Variables for the first component.

```
varCtr2 <- varCtr[,2]
names(varCtr2) <- rownames(varCtr)
a0006.Var.ctr2 <- PrettyBarPlot2(varCtr2,
main = 'Variable Contributions: Dimension 2', ylim = c(-.03, 1.2*max(varCtr2)),threshold = 1 / nrow(var
print(a0006.Var.ctr2)
```



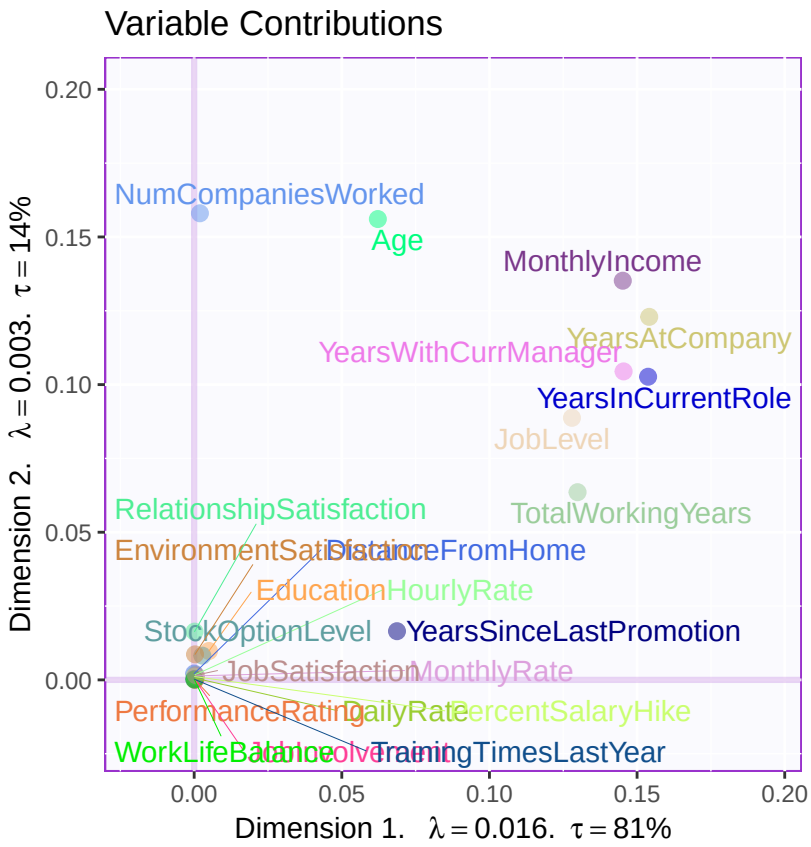

```
varCtr3 <- varCtr[,3]
names(varCtr3) <- rownames(varCtr)
a0006.Var.ctr3 <- PrettyBarPlot2(varCtr3,
main = 'Variable Contributions: Dimension 3', ylim = c(-.03, 1.2*max(varCtr2)),threshold = 1 / nrow(varC
)
print(a0006.Var.ctr3)
```

```
## Warning: Removed 2 rows containing missing values (position_stack).
```

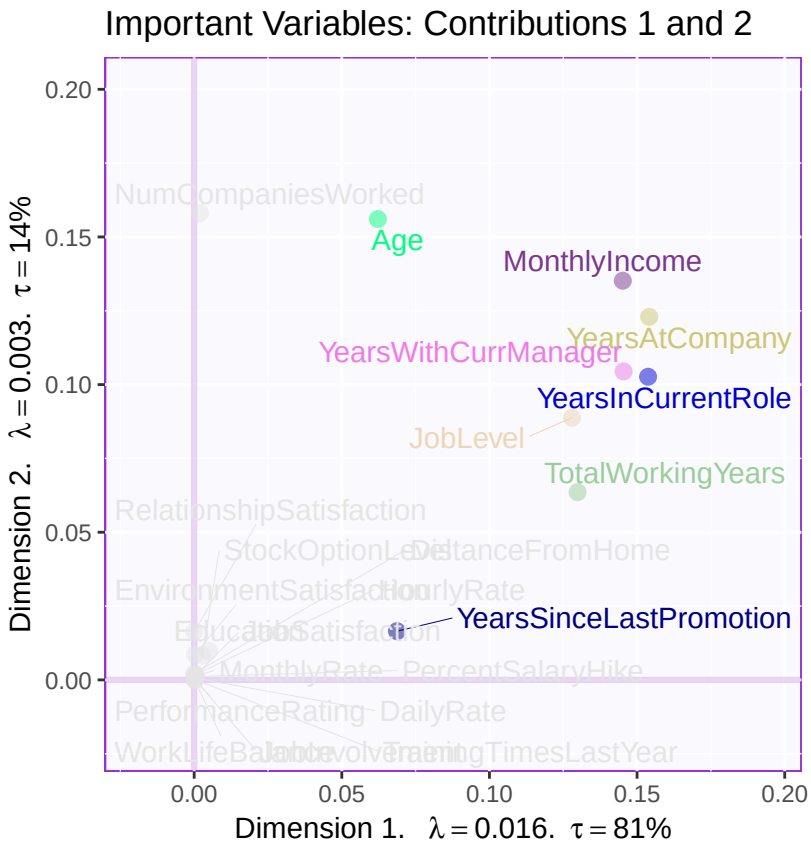


Pseudo factor plots with variable contributions

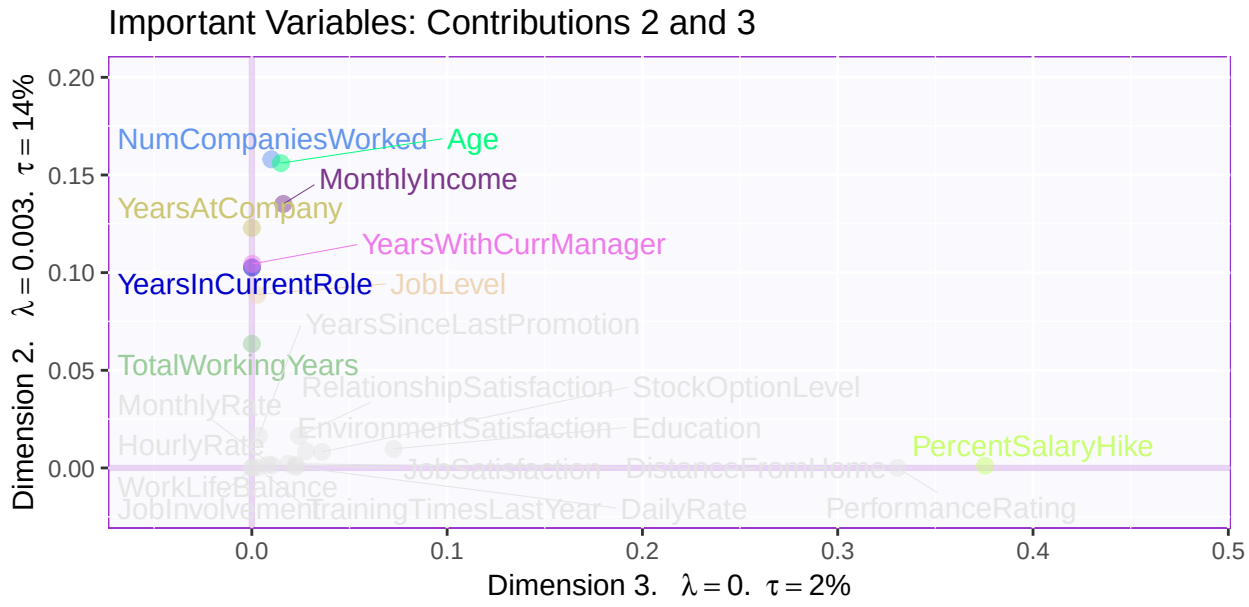
```
ctrV12 <- PTCA4CATA::createFactorMap(X = varCtr,
title = "Variable Contributions",
                        col.points = color,
                        col.labels = color,
                        alpha.points = 0.5,cex = 2.5,
                        alpha.labels = 1,
                        text.cex = 4,
                        font.face = "plain",
                        font.family = "sans")
ctr.labels <- createxyLabels.gen(
1,2, lambda = resMCA$ExPosition.Data$eigs, tau = resMCA$ExPosition.Data$t
)
a0007.Var.ctr12 <- ctrV12$zeMap + ctr.labels #
print(a0007.Var.ctr12)
```




```
absCtrVar <- as.matrix(varCtr) %*% diag(resMCA$ExPosition.Data$eigs)
varCtr12 <- (absCtrVar[,1] + absCtrVar[,2]) /
(resMCA$ExPosition.Data$eigs[1] + resMCA$ExPosition.Data$eigs[2])
importantVar <- (varCtr12 >= 1 / length(varCtr12))
col4ImportantVar <- color
col4NS <- 'gray90'
col4ImportantVar[!importantVar] <- col4NS
ctrV12.imp <- PTCA4CATA::createFactorMap(X = varCtr,
title = "Important Variables: Contributions 1 and 2",
                        col.points = col4ImportantVar,
                        col.labels = col4ImportantVar,
                        alpha.points = 0.5,
                        cex = 2.5,
                        alpha.labels = 1,
                        text.cex = 4,
                        font.face = "plain",
                        font.family = "sans")
a0008.Var.ctr12.imp <- ctrV12.imp$zeMap + ctr.labels #
print(a0008.Var.ctr12.imp)
```

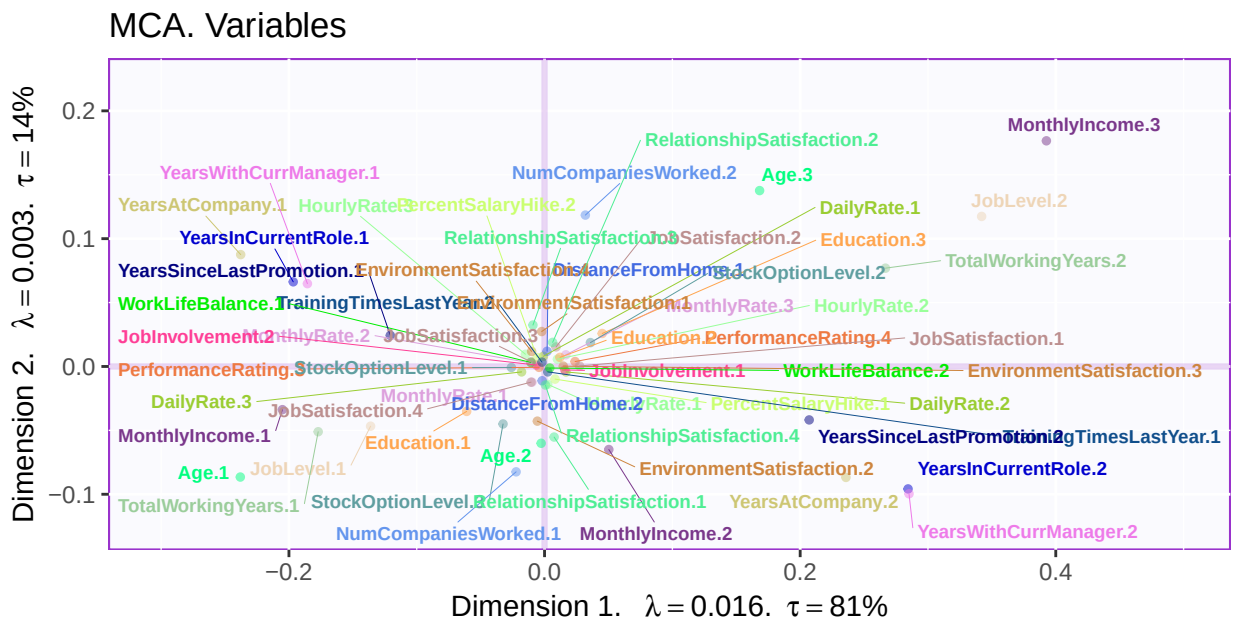



```
varCtr23 <- (absCtrVar[,3] + absCtrVar[,2]) / (resMCA$ExPosition.Data$eigs[3] + resMCA$ExPosition.Data$
importantVar23 <- (varCtr23 >= 1 / length(varCtr23))
col4ImportantVar23 <- color
col4NS <- 'gray90'
col4ImportantVar23[!importantVar23] <- col4NS
ctrV23.imp <- PTCA4CATA::createFactorMap(X = varCtr,
axis1 = 3, axis2 = 2,
                        title = "Important Variables: Contributions 2 and 3",
                        col.points = col4ImportantVar23,
                        col.labels = col4ImportantVar23,
                        alpha.points = 0.5,
                        cex = 2.5,
                        alpha.labels = 1,
                        text.cex = 4,
                        font.face = "plain",
                        font.family = "sans")
ctr.labels23 <- createxyLabels.gen(
3,2, lambda = resMCA$ExPosition.Data$eigs, tau = resMCA$ExPosition.Data$t
)
a0009.Var.ctr23.imp <- ctrV23.imp$zeMap + ctr.labels23 #
print(a0009.Var.ctr23.imp)
```

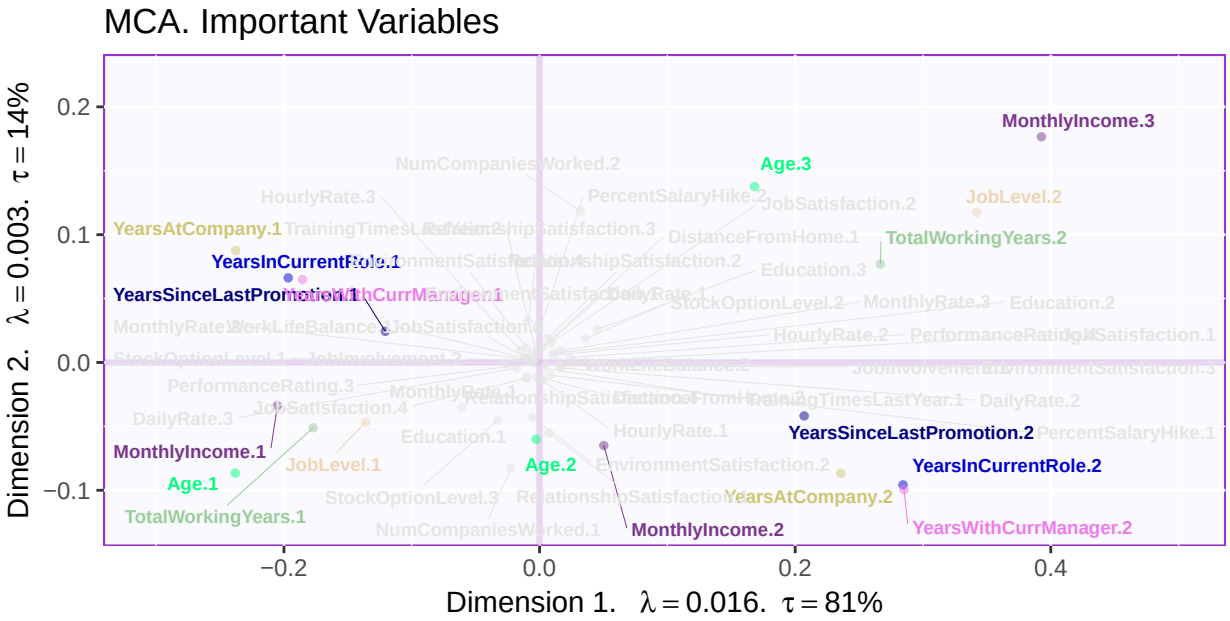


4.3 Variable Map for MCA

```
axis1 = 1
axis2 = 2
Fj1 <- resMCA$ExPosition.Data$fj
col4Levels <- data4PCCAR::coloringLevels( rownames(resMCA$ExPosition.Data$fj), color)
col4Labels <- col4Levels$color4Levels
# generate the set of maps
BaseMap.Fj <- createFactorMap(X = Fj1 , # resMCA$ExPosition.Data$fj,
                              axis1 = axis1, axis2 = axis2,
                              title = 'MCA. Variables',
                              col.points = col4Labels, cex = 1,
                              col.labels = col4Labels, text.cex = 2.5,
                              force = 2)
# add labels
labels4MCA <- createxyLabels.gen(x_axis = axis1, y_axis = axis2, lambda = resMCA$ExPosition.Data$eigs,
tau = resMCA$ExPosition.Data$t)
# make the maps
b0002.BaseMap.Fj <- BaseMap.Fj$zeMap + labels4MCA
b0003.BaseMapNoDot.Fj <- BaseMap.Fj$zeMap_background +
BaseMap.Fj$zeMap_text + labels4MCA
print(b0002.BaseMap.Fj)
```

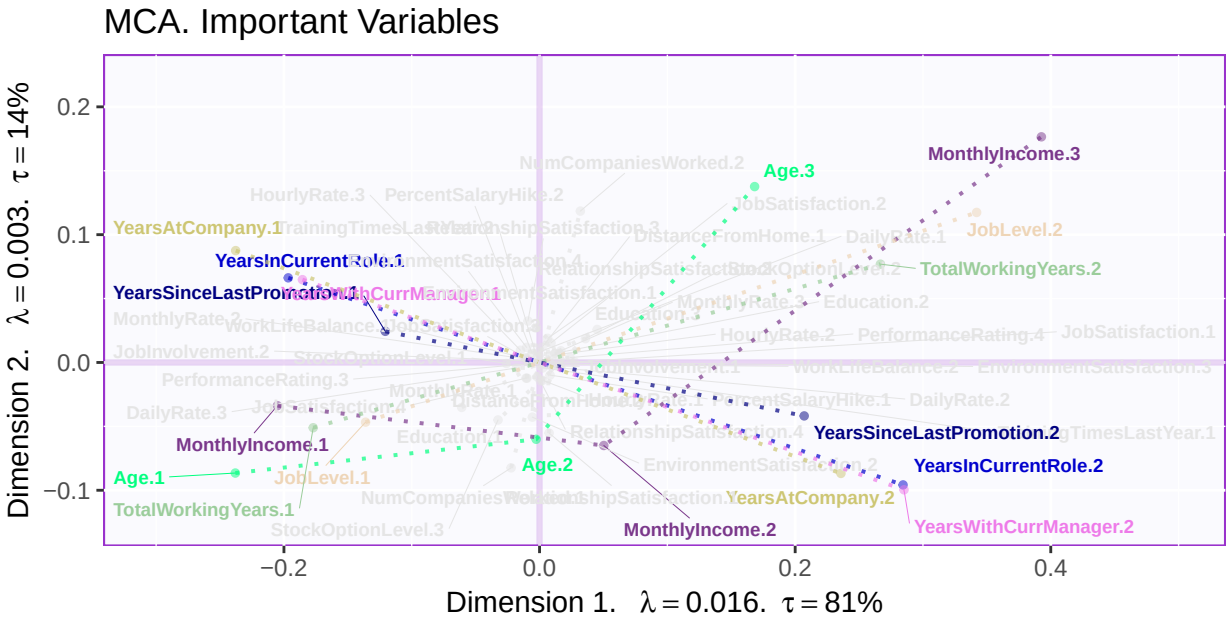



```
col4Levels.imp <- data4PCCAR::coloringLevels(rownames(Fj1), col4ImportantVar)
BaseMap.Fj.imp <- createFactorMap(X = Fj1 , # resMCA$ExPosition.Data$fj,
                                  axis1 = axis1, axis2 = axis2,
title = 'MCA. Important Variables', col.points = col4Levels.imp$color4Levels,
cex = 1,
col.labels = col4Levels.imp$color4Levels,
                                text.cex = 2.5,
force = 2)
b0010.BaseMap.Fj <- BaseMap.Fj.imp$zeMap + labels4MCA
print(b0010.BaseMap.Fj)
```



Map with important variables and lines

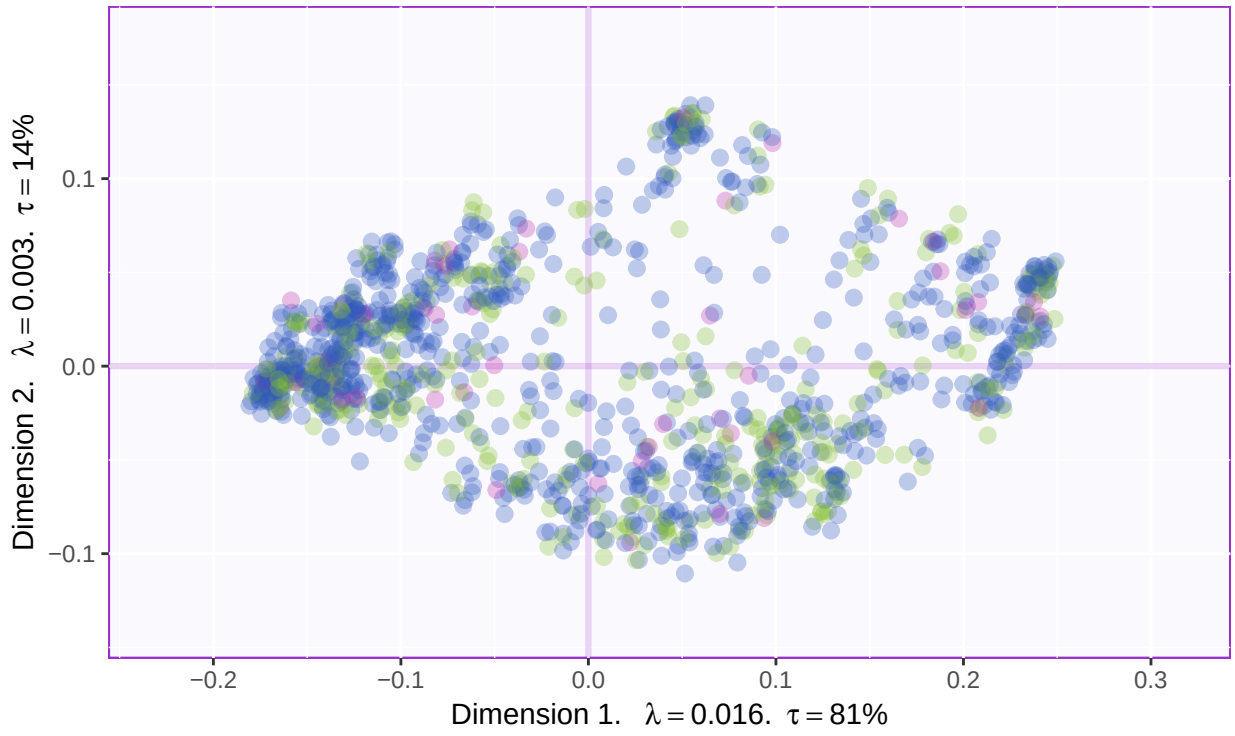
```
lines4J <- addLines4MCA(Fj1, col4Var = col4Levels.imp$color4Variables, size = .7)
b0020.BaseMap.Fj <- b0010.BaseMap.Fj + lines4J
print( b0020.BaseMap.Fj)
```



4.4 Factor Scores

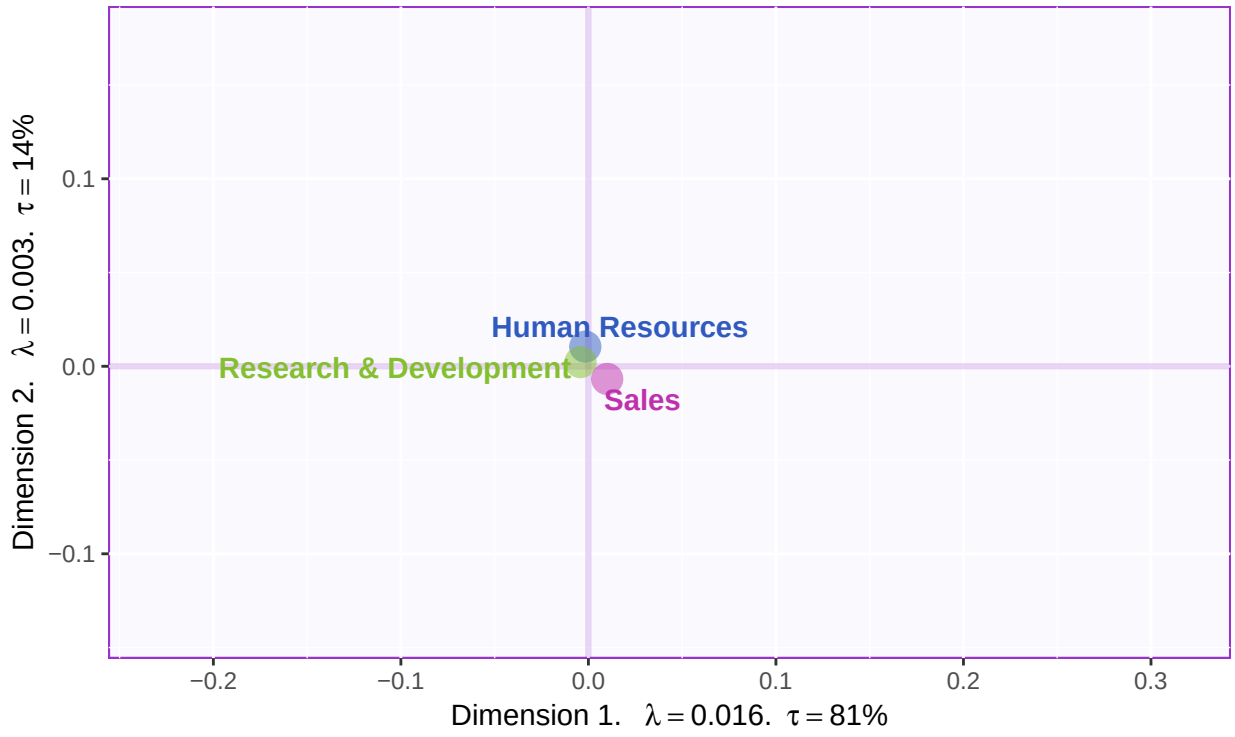
Department Type

```
baseMap.i1 <- PTCA4CATA::createFactorMap(resMCA$ExPosition.Data$fi,
                                         col.points   = resMCA$Plotting.Data$fi.col,
                                         alpha.points =  .3)
label4Map1 <- createxyLabels.gen(1,2,
                                 lambda = resMCA$ExPosition.Data$eigs,
                                 tau = resMCA$ExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i1 <- baseMap.i1$zeMap_background + baseMap.i1$zeMap_dots + label4Map1
#-----------------------------------------------------------------------
print(aggMap.i1)
```




```
col4data <- resMCA$Plotting.Data$fi.col
col4Means <- unique(col4data)
dataMeans <- getMeans(resMCA$ExPosition.Data$fi, my_data$Department)

MapGroup   <- PTCA4CATA::createFactorMap(dataMeans,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans <- baseMap.i1$zeMap_background  +
  MapGroup$zeMap_dots + MapGroup$zeMap_text + label4Map1
#------------------------------------------------------------------------
```

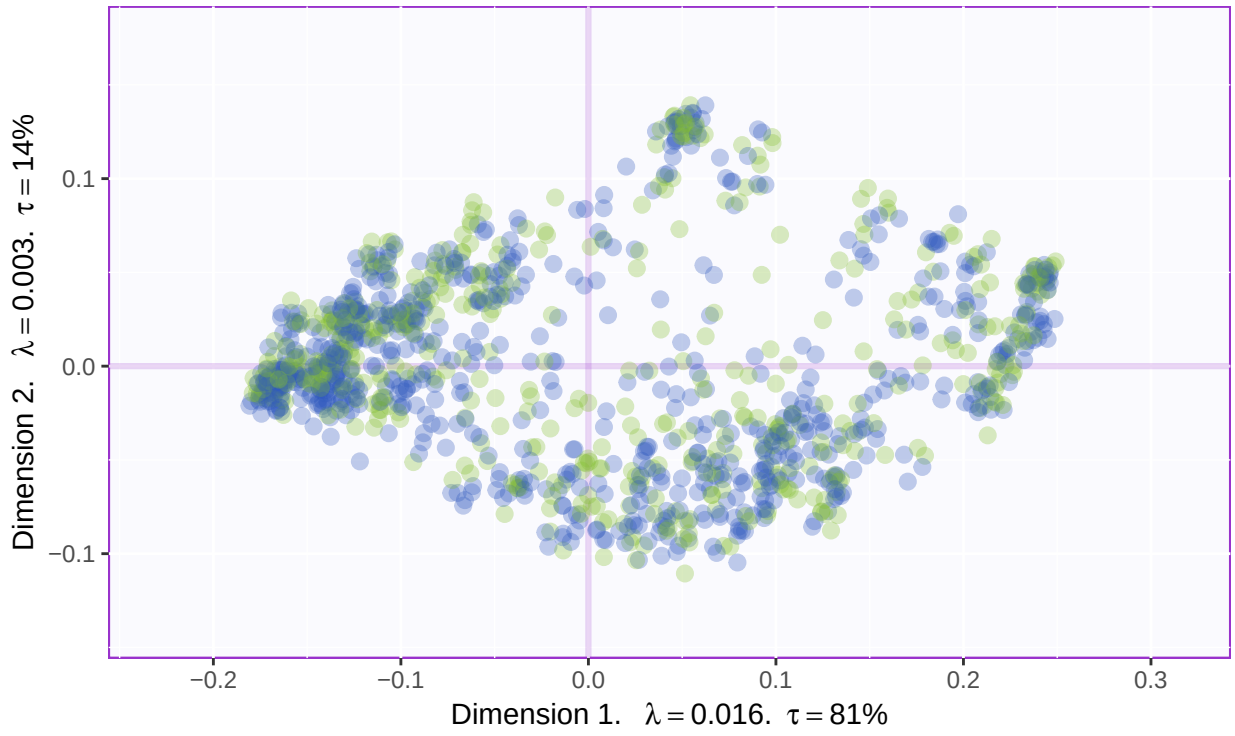
```
print(aggMap.i.withMeans)
```




```
BootCube.Gr <- PTCA4CATA::Boot4Mean(resMCA$ExPosition.Data$fi, design = my_data$Department,
                                    niter = 100,
                                    suppressProgressBar = TRUE)

GraphElli <- MakeCIEllipses(BootCube.Gr$BootCube[,1:2,],
                  names.of.factors = c("Dimension 1","Dimension 2"),
                  col = col4Means,
                  p.level = .95
)


aggMap.i.withCI2 <- baseMap.i1$zeMap_background +  GraphElli + MapGroup$zeMap_text + MapGroup$zeMap_dots

print(aggMap.i.withCI2)
```

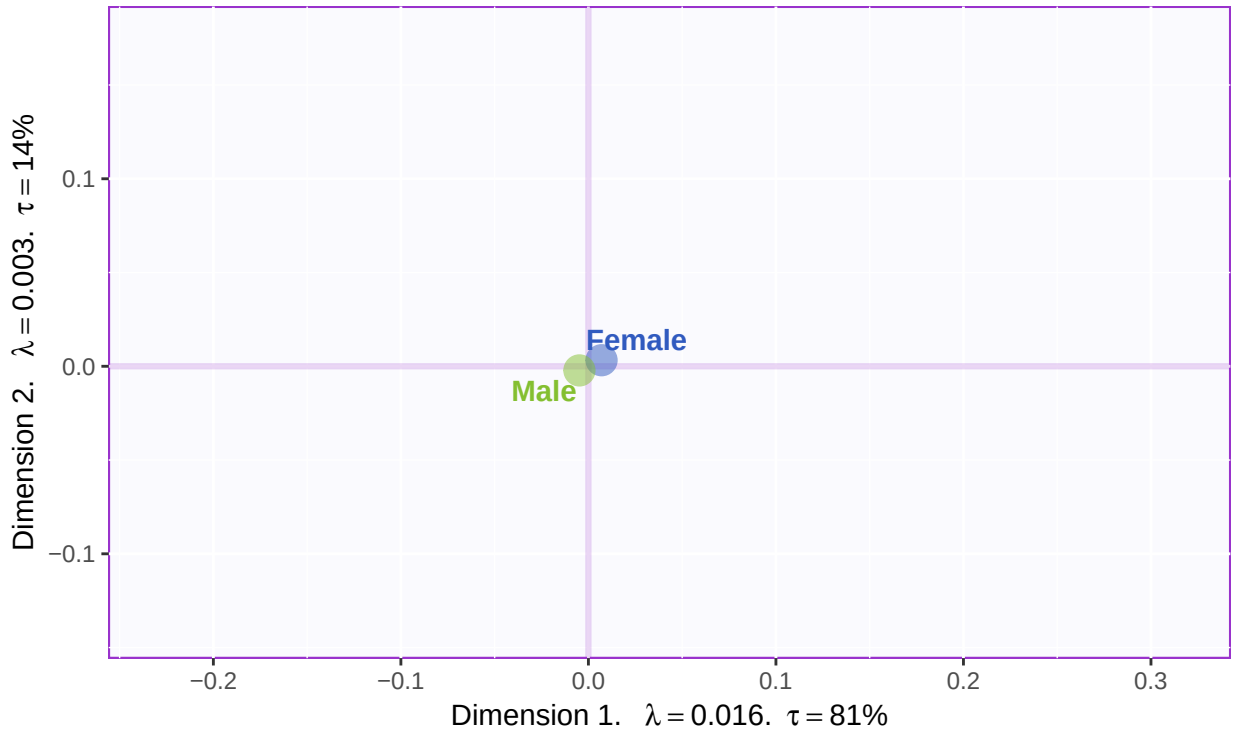
Gender Type

```
baseMap.i2 <- PTCA4CATA::createFactorMap(resMCA1$ExPosition.Data$fi,
                                         col.points   = resMCA1$Plotting.Data$fi.col,
                                         alpha.points =  .3)
label4Map2 <- createxyLabels.gen(1,2,
                                 lambda = resMCA1$ExPosition.Data$eigs,
                                 tau = resMCA1$ExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i2 <- baseMap.i2$zeMap_background + baseMap.i2$zeMap_dots + label4Map2
#-----------------------------------------------------------------------
print(aggMap.i2)
```

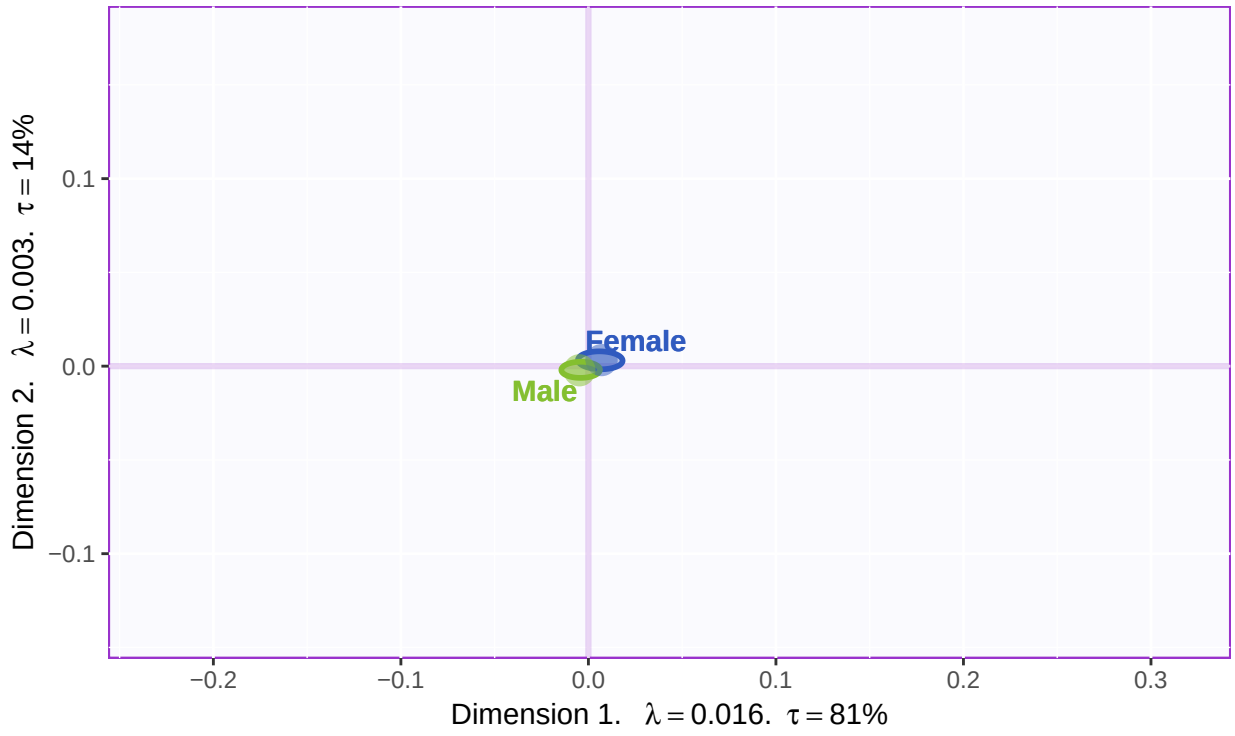



```
col4data1 <- resMCA1$Plotting.Data$fi.col
col4Means1 <- unique(col4data1)
dataMeans1 <- getMeans(resMCA1$ExPosition.Data$fi, my_data$Gender)

MapGroup1   <- PTCA4CATA::createFactorMap(dataMeans1,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means1,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means1,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans1 <- baseMap.i2$zeMap_background +
  MapGroup1$zeMap_dots + MapGroup1$zeMap_text + label4Map2
#------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans1)
```

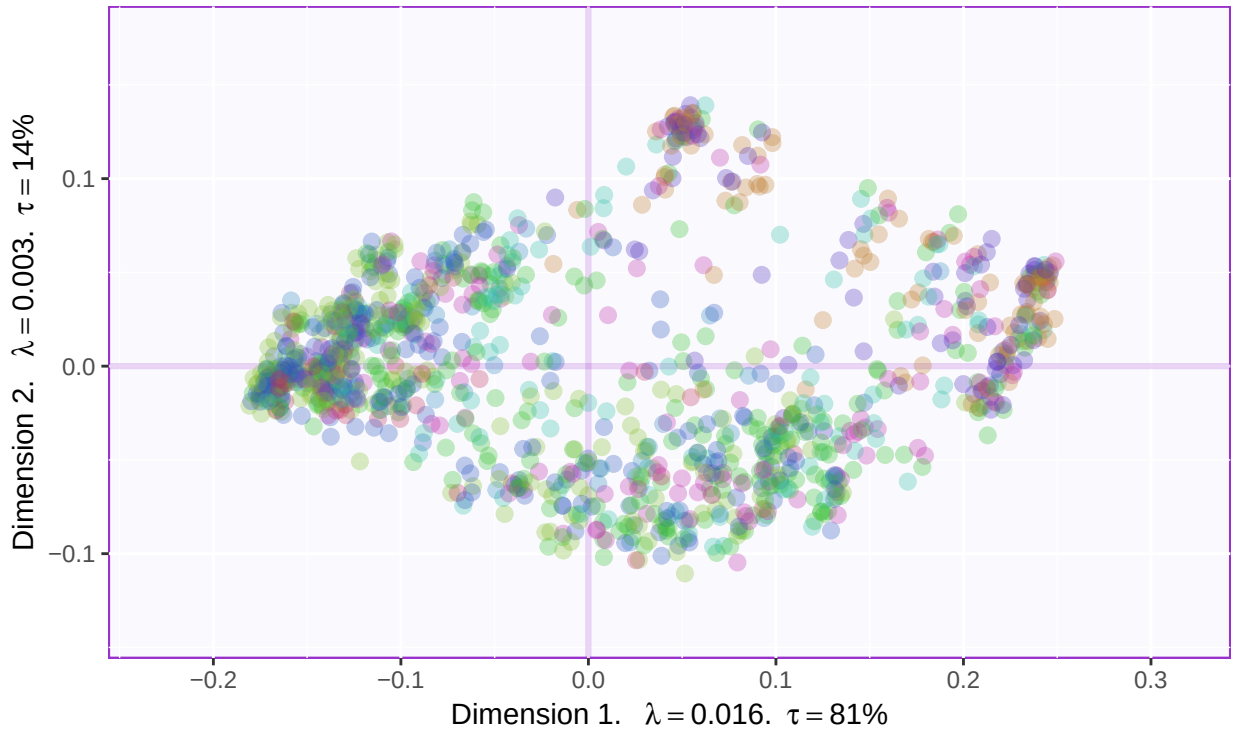



```
BootCube.Gr1 <- PTCA4CATA::Boot4Mean(resMCA1$ExPosition.Data$fi, design = my_data$Gender,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli1 <- MakeCIEllipses(BootCube.Gr1$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means1,
                   p.level = .95
)


aggMap.i.withCI1 <- baseMap.i2$zeMap_background +  GraphElli1 + MapGroup1$zeMap_text + MapGroup1$zeMap_

print(aggMap.i.withCI1)
```



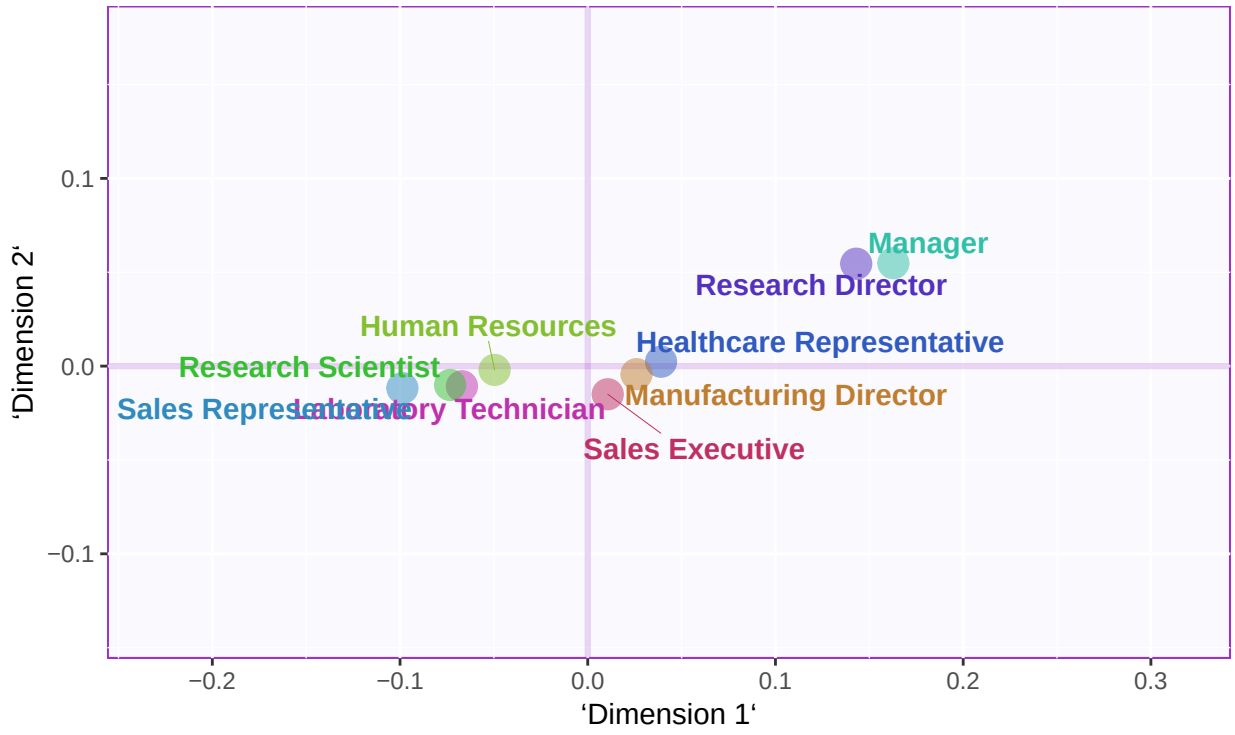
Job Role

```
resMCA2 <- epMCA(data1 ,make_data_nominal = TRUE ,DESIGN = my_data$JobRole ,make_design_nominal = TRUE,g

baseMap.i22 <- PTCA4CATA::createFactorMap(resMCA2$ExPosition.Data$fi,
                                          col.points   = resMCA2$Plotting.Data$fi.col,
                                          alpha.points =  .3)
label4Map22 <- createxyLabels.gen(1,2,
                                  lambda = resMCA2$ExPosition.Data$eigs,
                                  tau = resMCA2$ExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i12 <- baseMap.i22$zeMap_background + baseMap.i22$zeMap_dots + label4Map22
#------------------------------------------------------------------------
print(aggMap.i12)
```

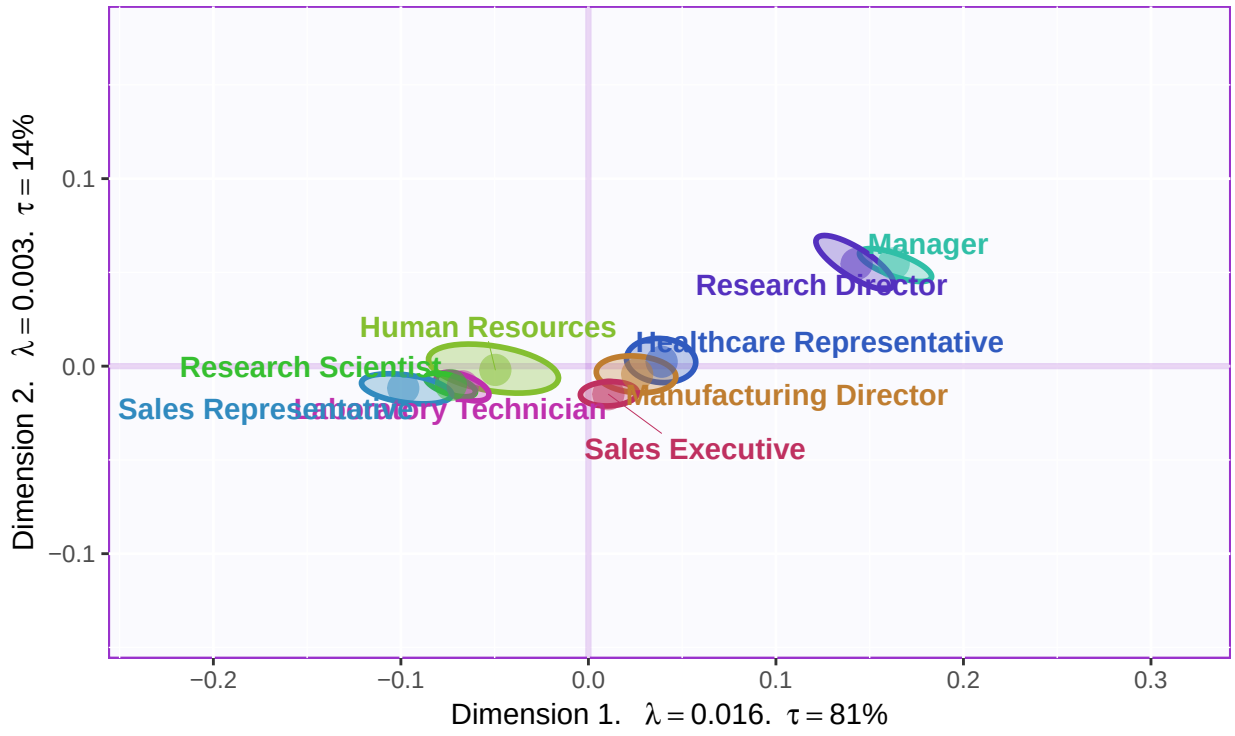



```
col4data2 <- resMCA2$Plotting.Data$fi.col
col4Means2 <- unique(col4data2)
dataMeans2 <- getMeans(resMCA2$ExPosition.Data$fi, my_data$JobRole)

MapGroup2   <- PTCA4CATA::createFactorMap(dataMeans2,
                                axis1 = 1, axis2 = 2,
                                col.points = col4Means2,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means2,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1, title = "Mean CI for JobRole",
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans12 <- aggMap.i12 +
  MapGroup2$zeMap_dots + MapGroup2$zeMap_text
#-----------------------------------------------------------------------
ab <- baseMap.i22$zeMap_background +
  MapGroup2$zeMap_dots + MapGroup2$zeMap_text
```

```
print(ab)
```

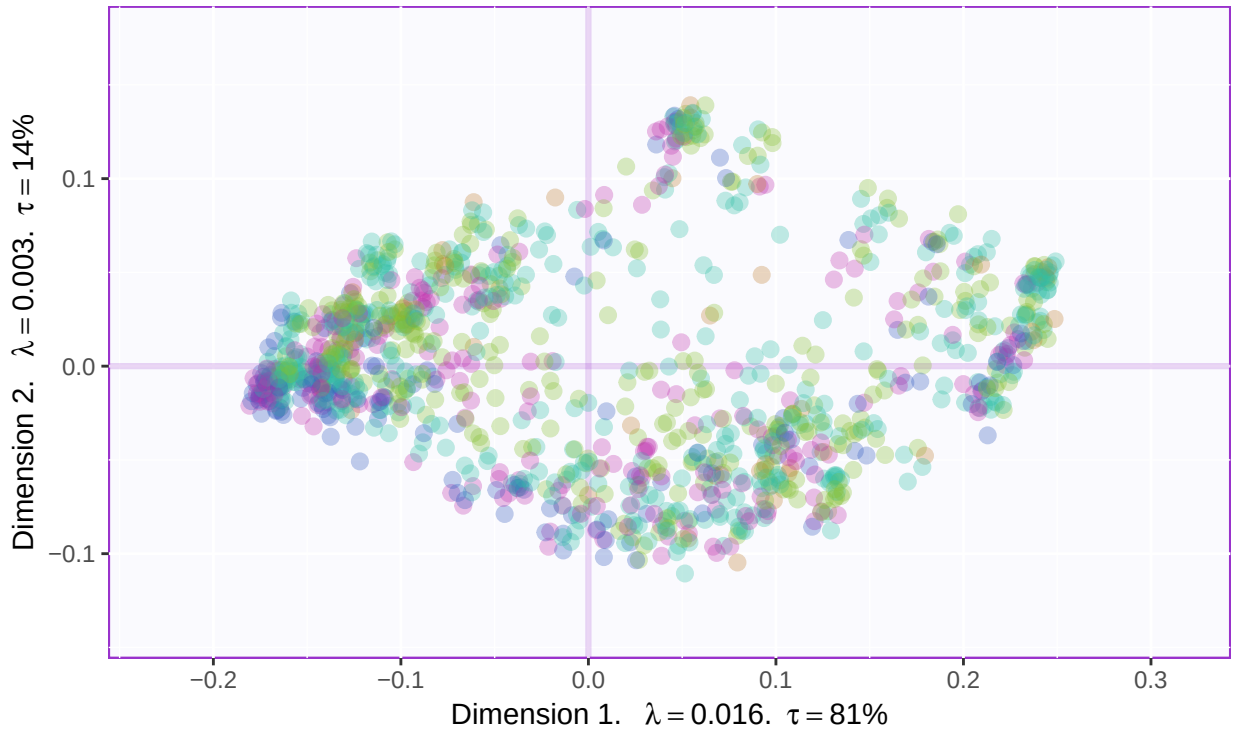



```
BootCube.Gr2 <- PTCA4CATA::Boot4Mean(resMCA2$ExPosition.Data$fi, design = my_data$JobRole,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli2 <- MakeCIEllipses(BootCube.Gr2$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means2,
                   p.level = .95
)


aggMap.i.withCI22 <- baseMap.i22$zeMap_background +  GraphElli2 + MapGroup2$zeMap_text + MapGroup2$zeMa

print(aggMap.i.withCI22)
```



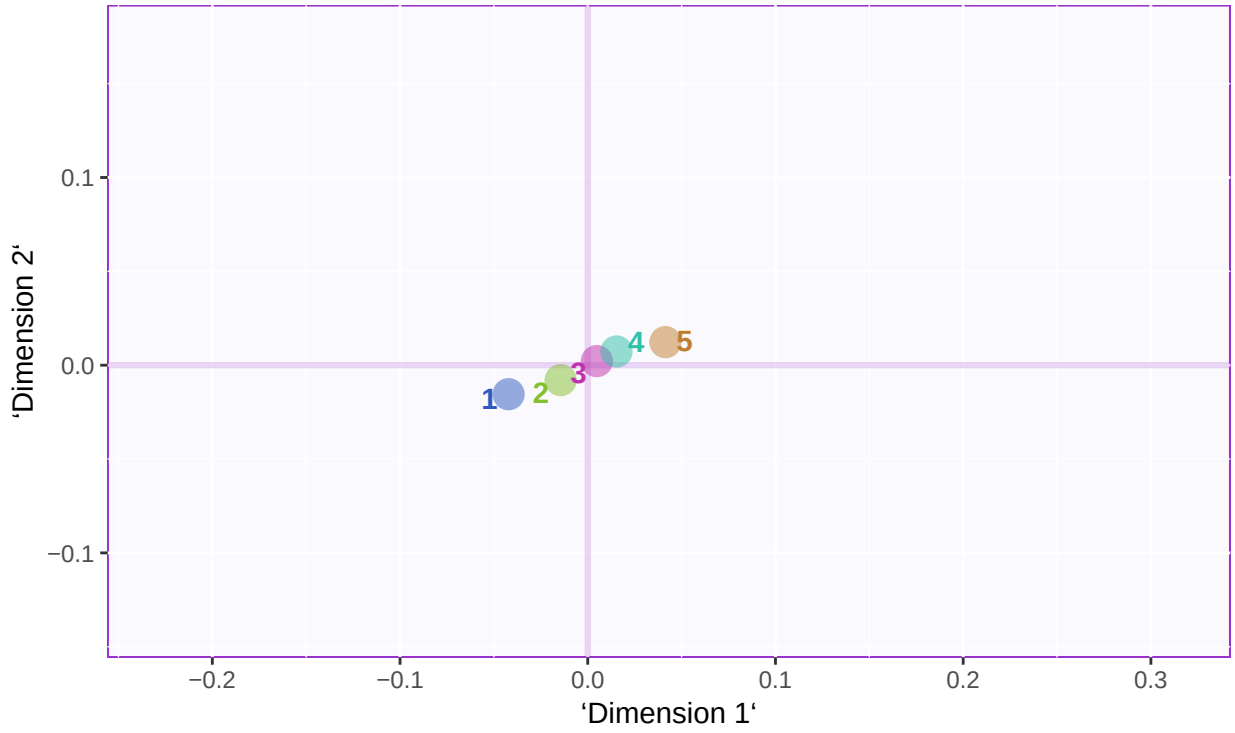
Education

```
resMCA3 <- epMCA(data1 ,make_data_nominal = TRUE ,DESIGN = my_data$Education ,make_design_nominal = TRU

baseMap.i221 <- PTCA4CATA::createFactorMap(resMCA3$ExPosition.Data$fi,
                                           col.points    = resMCA3$Plotting.Data$fi.col,
                                           alpha.points =  .3)
label4Map221 <- createxyLabels.gen(1,2,
                                   lambda = resMCA3$ExPosition.Data$eigs,
                                   tau = resMCA3$ExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i3 <- baseMap.i221$zeMap_background + baseMap.i221$zeMap_dots + label4Map221
#-----------------------------------------------------------------------
print(aggMap.i3)
```

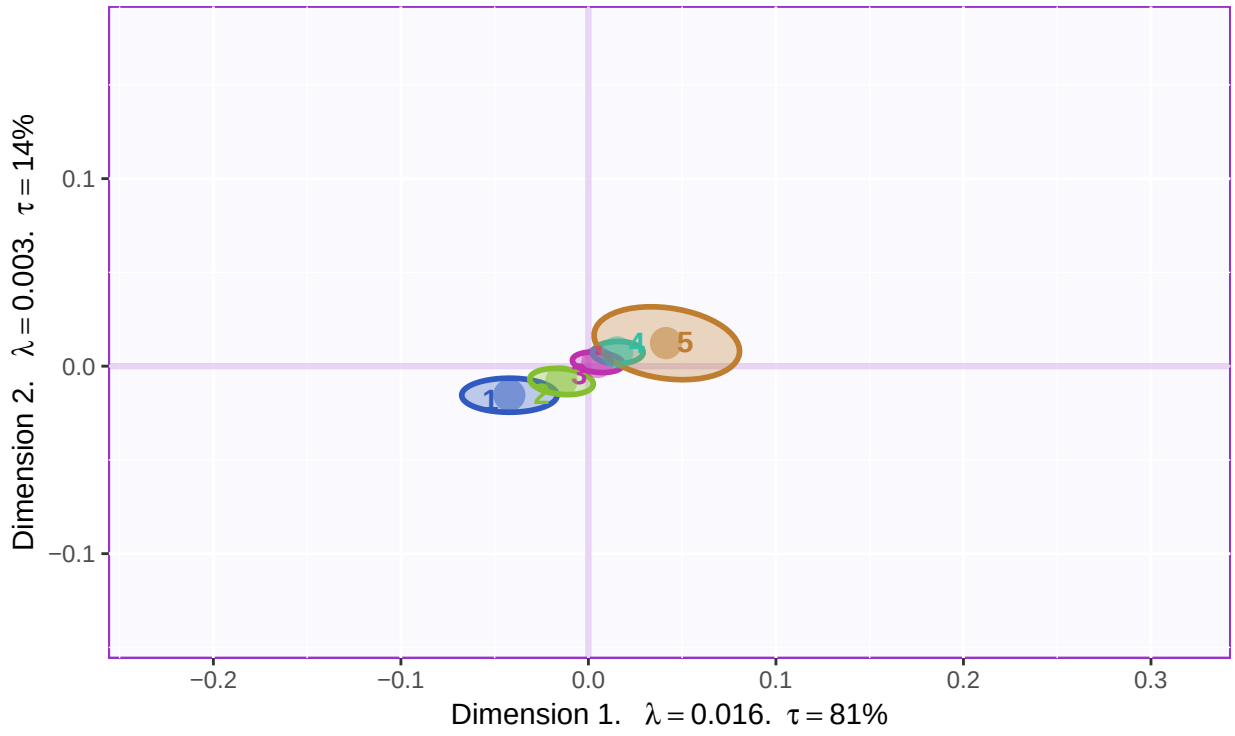



```
col4data3 <- resMCA3$Plotting.Data$fi.col
col4Means3 <- unique(col4data3)
dataMeans3 <- getMeans(resMCA3$ExPosition.Data$fi, my_data$Education)

MapGroup3   <- PTCA4CATA::createFactorMap(dataMeans3,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means3,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means3,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans3 <- aggMap.i3 +
  MapGroup3$zeMap_dots + MapGroup3$zeMap_text
#-----------------------------------------------------------------------
```

```
abc <- baseMap.i221$zeMap_background +
  MapGroup3$zeMap_dots + MapGroup3$zeMap_text

print(abc)
```

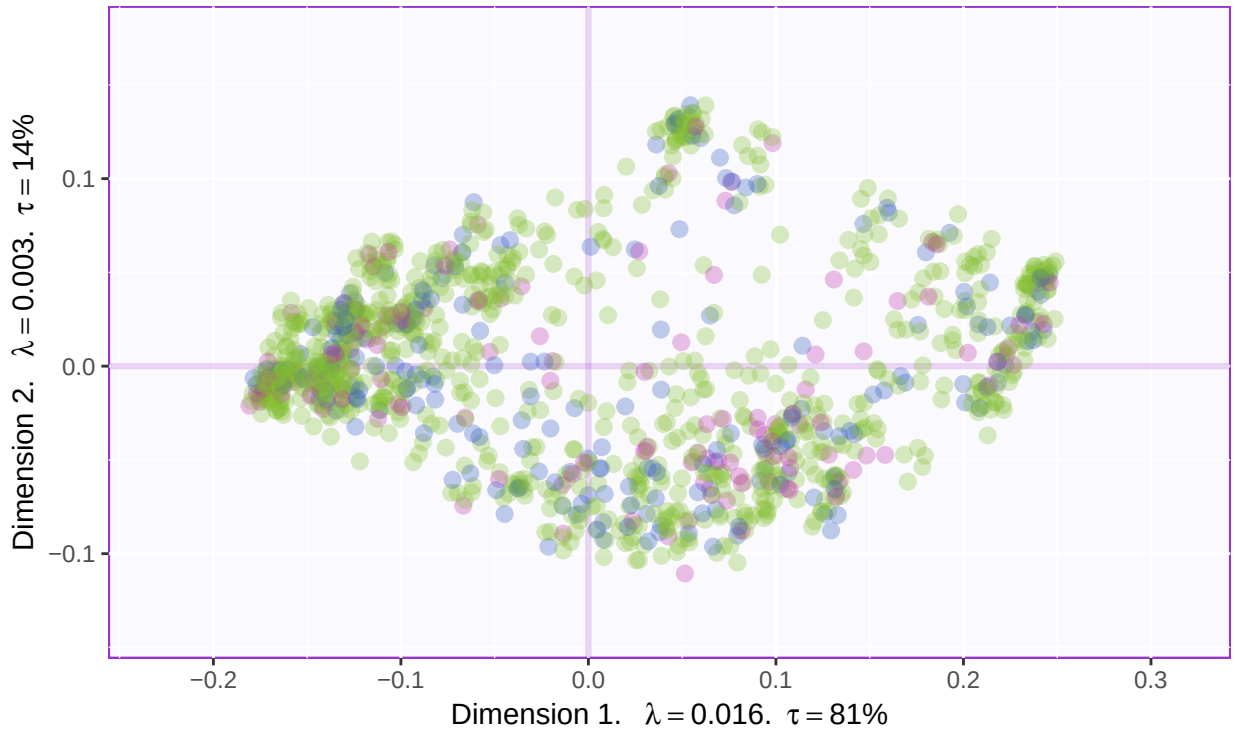



```
BootCube.Gr3 <- PTCA4CATA::Boot4Mean(resMCA3$ExPosition.Data$fi, design = my_data$Education,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli3 <- MakeCIEllipses(BootCube.Gr3$BootCube[,1:2,],
                  names.of.factors = c("Dimension 1","Dimension 2"),
                  col = col4Means3,
                  p.level = .95
)


aggMap.i.withCI3 <- baseMap.i221$zeMap_background+  GraphElli3 + MapGroup3$zeMap_text + MapGroup3$zeMap_

print(aggMap.i.withCI3)
```



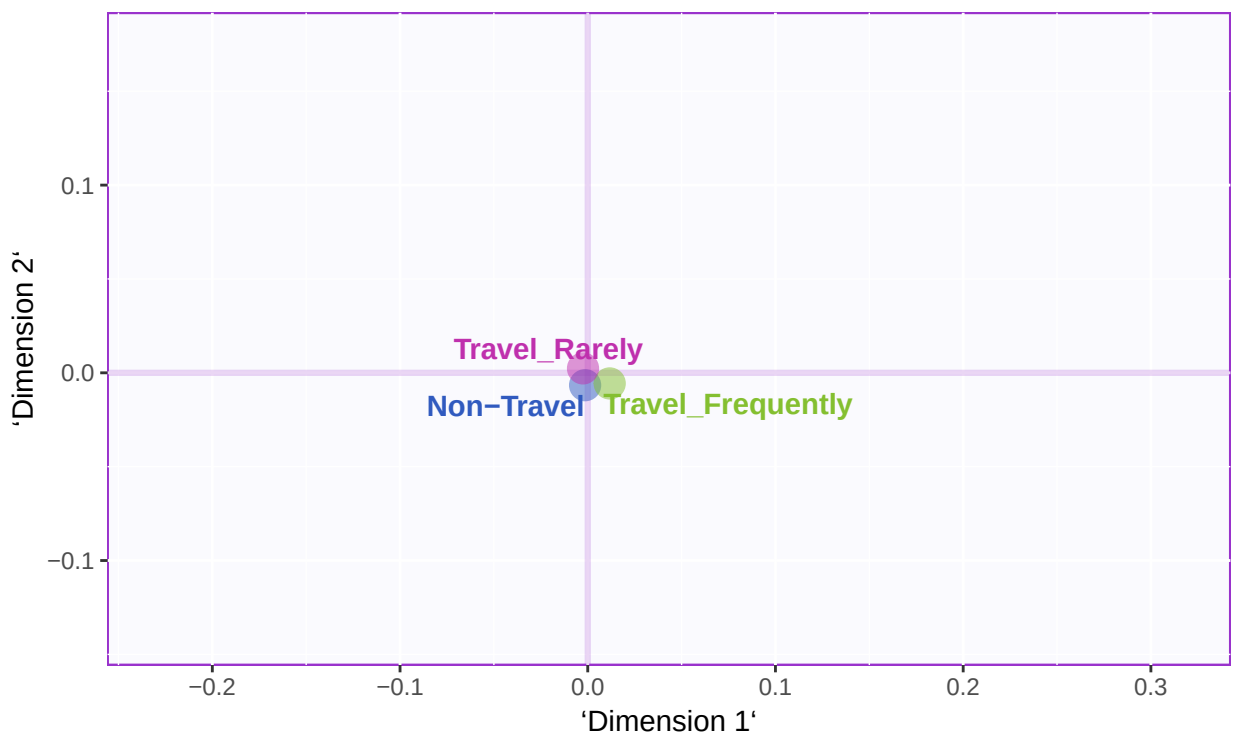
Travel

```
resMCA4 <- epMCA(data1 ,make_data_nominal = TRUE ,DESIGN = my_data$BusinessTravel ,make_design_nominal =

baseMap.i24 <- PTCA4CATA::createFactorMap(resMCA4$ExPosition.Data$fi,
                                          col.points    = resMCA4$Plotting.Data$fi.col,
                                          alpha.points =  .3)
label4Map24 <- createxyLabels.gen(1,2,
                                  lambda = resMCA4$ExPosition.Data$eigs,
                                  tau = resMCA4$ExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i14 <- baseMap.i24$zeMap_background + baseMap.i24$zeMap_dots + label4Map24
#------------------------------------------------------------------------
print(aggMap.i14)
```

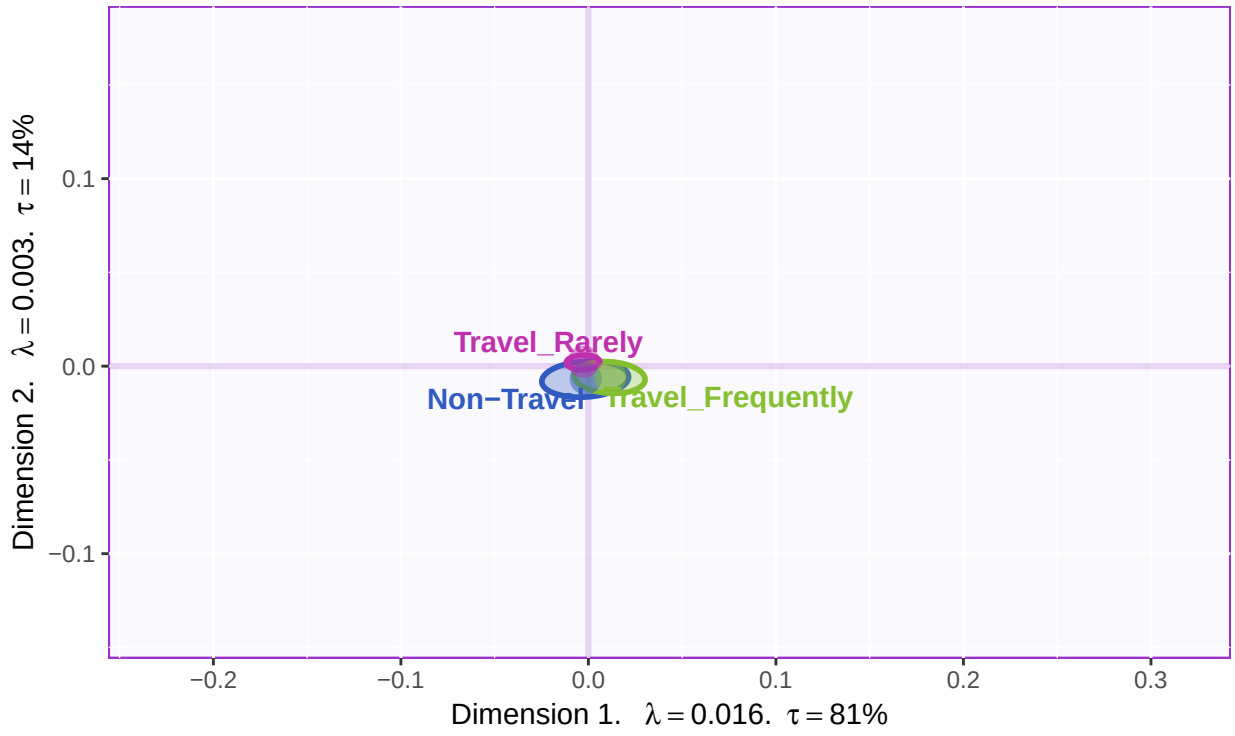



```
col4data4 <- resMCA4$Plotting.Data$fi.col
col4Means4 <- unique(col4data4)
dataMeans4 <- getMeans(resMCA4$ExPosition.Data$fi, my_data$BusinessTravel)

MapGroup4   <- PTCA4CATA::createFactorMap(dataMeans4,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means4,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means4,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans14 <- aggMap.i14 +
  MapGroup4$zeMap_dots + MapGroup4$zeMap_text
#-----------------------------------------------------------------------
```

```
ab4 <- baseMap.i24$zeMap_background +
  MapGroup4$zeMap_dots + MapGroup4$zeMap_text
print(ab4)
```



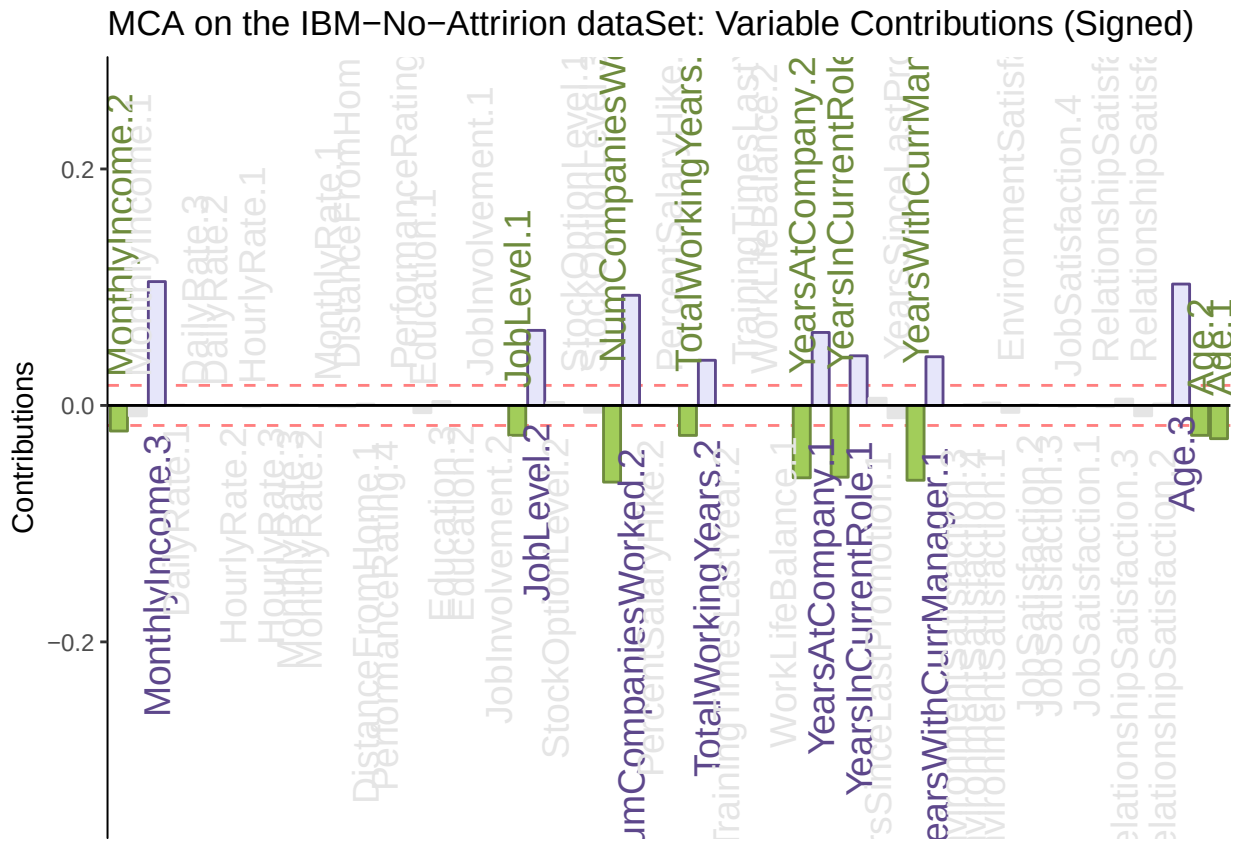

```
BootCube.Gr4 <- PTCA4CATA::Boot4Mean(resMCA4$ExPosition.Data$fi, design = my_data$BusinessTravel,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli4 <- MakeCIEllipses(BootCube.Gr4$BootCube[,1:2,],
                  names.of.factors = c("Dimension 1","Dimension 2"),
                  col = col4Means4,
                  p.level = .95
)


aggMap.i.withCI24 <- baseMap.i24$zeMap_background +  GraphElli4 + MapGroup4$zeMap_text + MapGroup4$zeMap

print(aggMap.i.withCI24)
```



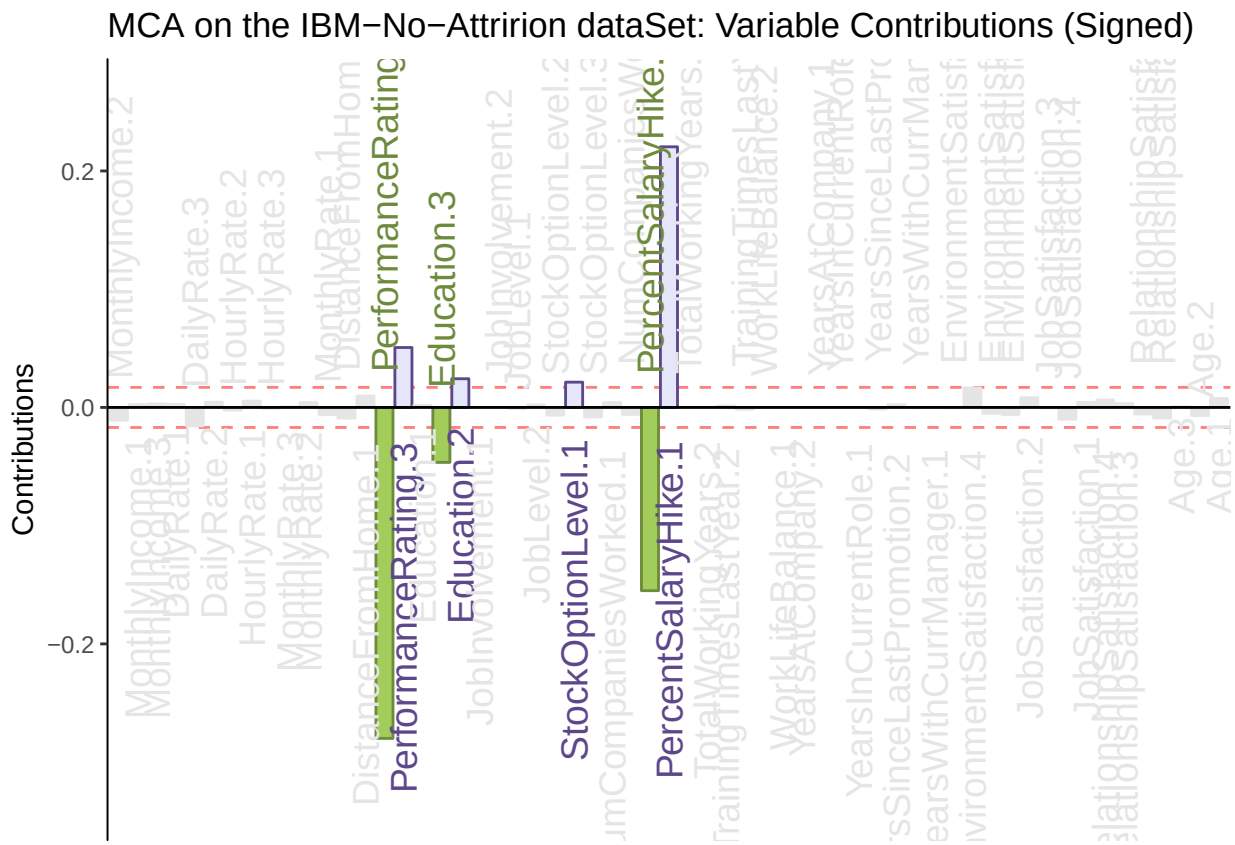
4.5 Contribution for variables

```
signed.ctrJ2 <- resMCA$ExPosition.Data$cj * sign(resMCA$ExPosition.Data$fj)
b003.ctrJ.s.111 <- PrettyBarPlot2(signed.ctrJ2[,1],
                                  threshold = 1 / NROW(signed.ctrJ2),
                                  font.size = 5,
                                 # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                  main = 'MCA on the IBM-No-Attririon data Set: Variable Contributions (S
                                  ylab = 'Contributions',
                                  ylim = c(1.2*min(signed.ctrJ2), 1.2*max(signed.ctrJ2))
)
print(b003.ctrJ.s.111)
```


```
b004.ctrJ.s.211 <- PrettyBarPlot2(signed.ctrJ2[,2],
                                  threshold = 1 / NROW(signed.ctrJ2),
                                  font.size = 5,
                                 # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                  main = 'MCA on the IBM-No-Attririon dataSet: Variable Contributions (Si
                                  ylab = 'Contributions',
                                  ylim = c(1.2*min(signed.ctrJ2), 1.2*max(signed.ctrJ2))
)
print(b004.ctrJ.s.211)
```

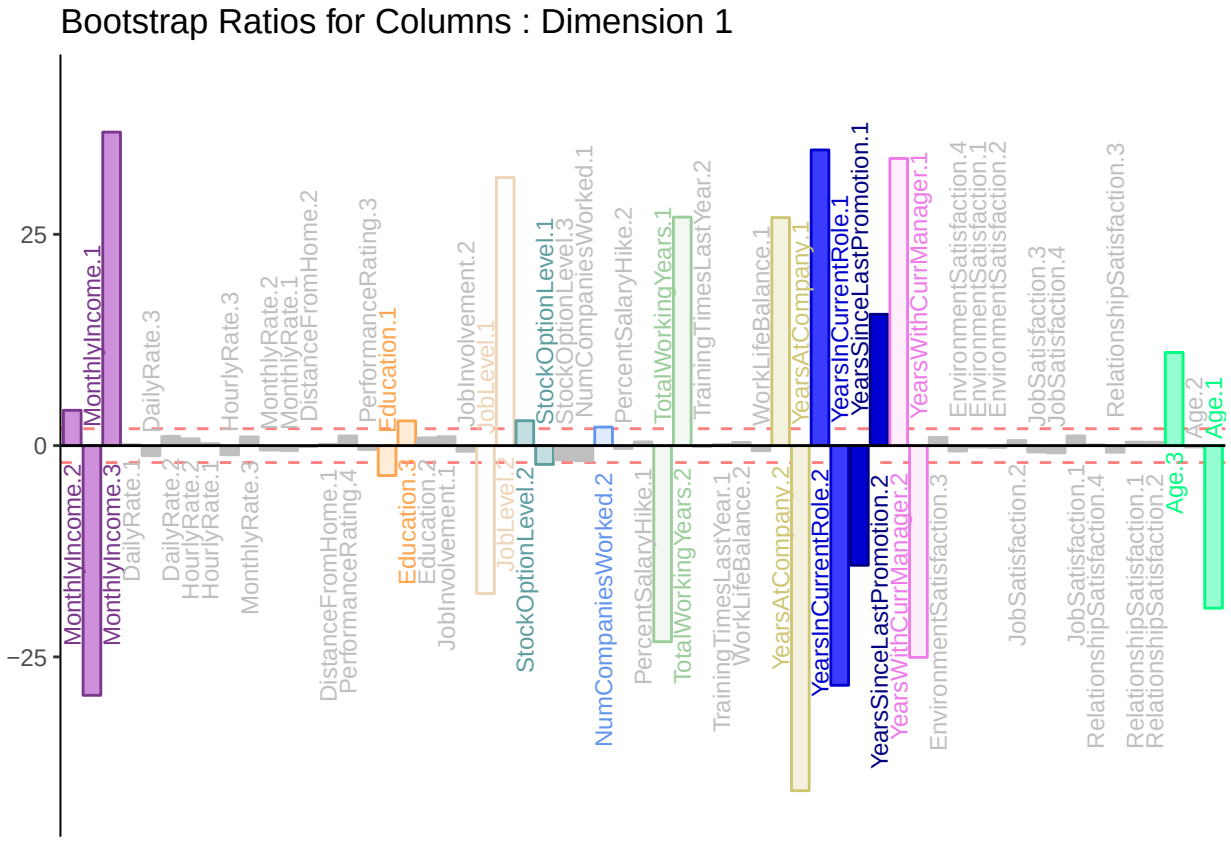



```
b004.ctrJ.s.311 <- PrettyBarPlot2(signed.ctrJ2[,3],
                                  threshold = 1 / NROW(signed.ctrJ2),
                                  font.size = 5,
                                 # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                  main = 'MCA on the IBM-No-Attririon dataSet: Variable Contributions (Si
                                  ylab = 'Contributions',
                                  ylim = c(1.2*min(signed.ctrJ2), 1.2*max(signed.ctrJ2))
)
print(b004.ctrJ.s.311)
```

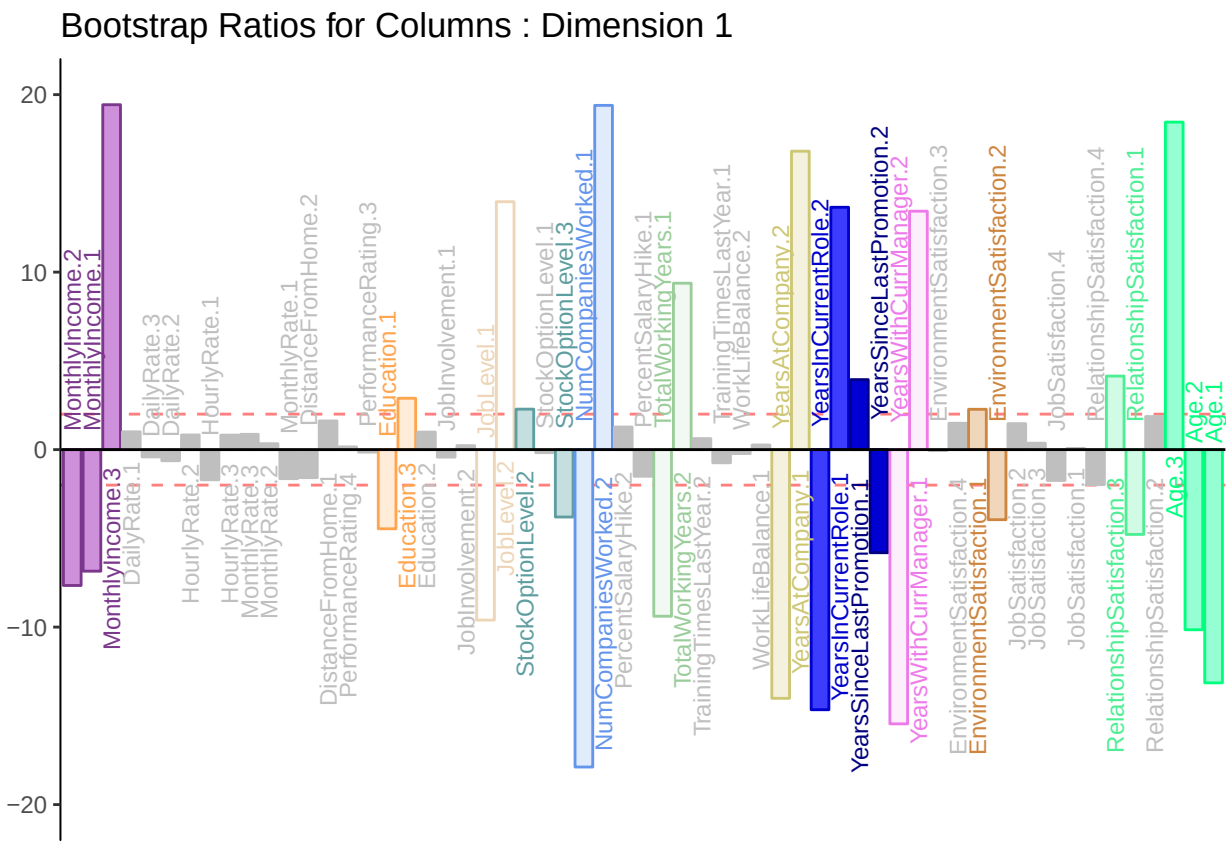


4.6 Bootstrap Ratios for Variables

```
col4Labels <- col4Levels$color4Levels
c0001.Levels.BR <- PrettyBarPlot2( resMCA.inf$Inference.Data$fj.boots$tests$boot.ratios[,1], # BR
                    main = 'Bootstrap Ratios for Columns : Dimension 1',
                             threshold = 2,color4bar = gplots::col2hex(col4Labels)
)
c0001.Levels.BR2 <- PrettyBarPlot2( resMCA.inf$Inference.Data$fj.boots$tests$boot.ratios[,2], # BR
                    main = 'Bootstrap Ratios for Columns : Dimension 1',
                             threshold = 2,color4bar = gplots::col2hex(col4Labels)
)
print(c0001.Levels.BR)
```




```
print(c0001.Levels.BR2)
```



Summary

When we interpret the factor scores and loadings together, the MCA revealed almost the similar results as PCA:

- Usually the Research and Development people have more work-experience & Monthly Income in comparison to other department type
- Generally, Managers and Directors are the ones with PhD & Master's with higher job level, Sales representative and Lab Technician have no college education while Healthcare Representatives & Manufacturing Directors have Bachelor's degree
- People who are Managers and Research Directors have the highest pay and are more seasoned while people who are Sales representative and Lab Technician have low pay and less experience
- It is also observed that the Mangers and Research Directors travel the most, could be their meetings and conferences while the HR, Sales representative,Lab Technician, Research Scientists travel occasionally
- Also, we can say here that Age is also an important factor in the JOb level i.e Lower the position younger the age and higher the position the more seasoned the employees are.
- For loadings we see that Monthly Income, Job Level, Number of Years of Experience ,Age are the important variables for the first component and they all follow the non-linear relationship with the data.
- For the factor plot the gender type does not show any significant inference as the mean are almost overlapping each other and for the department type similar results can be seen as R&D, Sales and HR all intersect their mean confidence Intervals.

```
options(knitr.duplicate.label = 'allow')
```

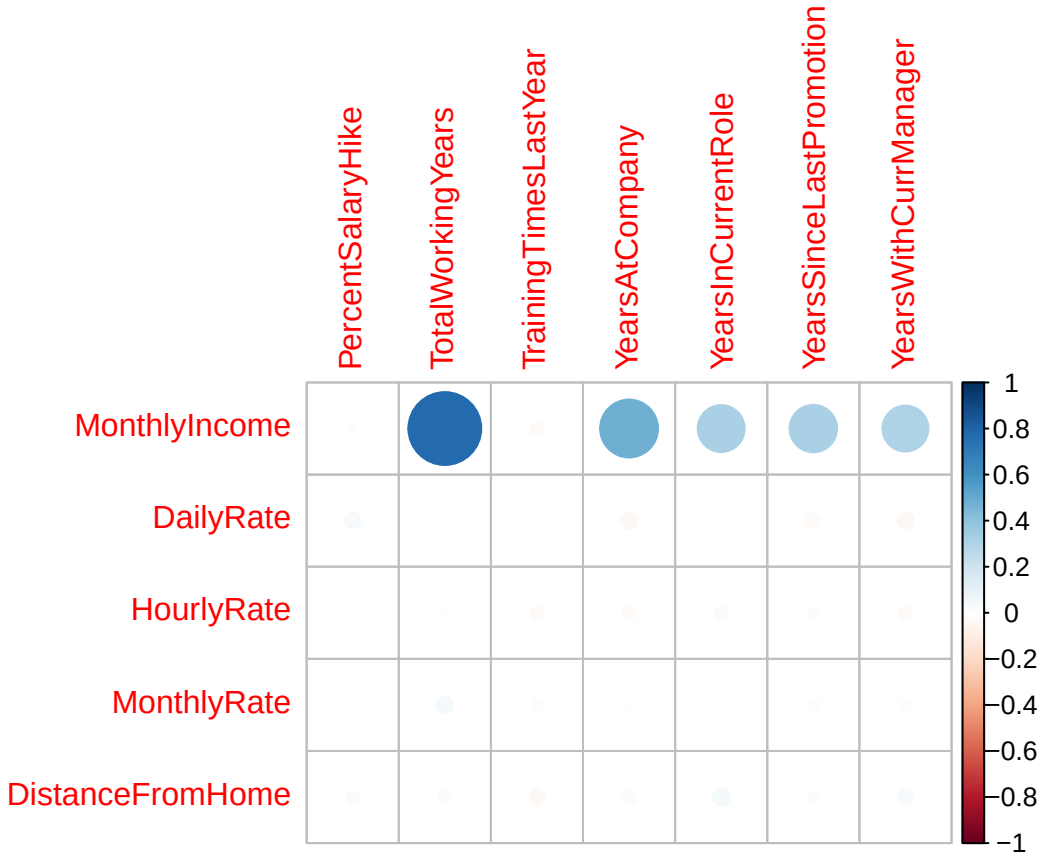
5 PLS

Partial least square (PLS) methods (also sometimes called projection to latent structures) relate the information present in two data tables that collect measurements on the same set of observations. PLS methods proceed by deriving latent variables which are (optimal) linear combinations of the variables of a data table. When the goal is to find the shared information between two tables, the approach is equivalent to a correlation problem and the technique is then called partial least square correlation (PLSC) (also sometimes called PLS-SVD). In this case there are two sets of latent variables (one set per table), and these latent variables are required to have maximal covariance.

```
options(knitr.duplicate.label = 'allow')
data <- read.csv("IBM-HR-Emplyee-NoAttrition.csv")
data1 <- data[,11:15]
data2 <- data[,22:29]
data2 <- data2[ -c(4) ]
#data$Gender <- as.numeric(as.factor(data$Gender))
#data$Department <- as.numeric(as.factor(data$Department))
#data$JobRole <- as.numeric(as.factor(data$JobRole))
#data$EducationField <- as.numeric(as.factor(data$EducationField))
design1 <- data$Gender
design2 <- data$Department
design3 <- data$JobRole
design4 <- data$Education
design5 <- data$BusinessTravel
```

Correlation Plot

```
corrplot::corrplot(cor(data1,data2))
```



From the correlation plot we can clearly see that the Monthly Income is correlated with the Years of experience.We don't see any relation of Montlhy Rate,DailyRate and Hourly Rate with the year's of experience because the scatter plot for these variables are scattered uniformly all over the x and y axis telling that their is zero correlation.

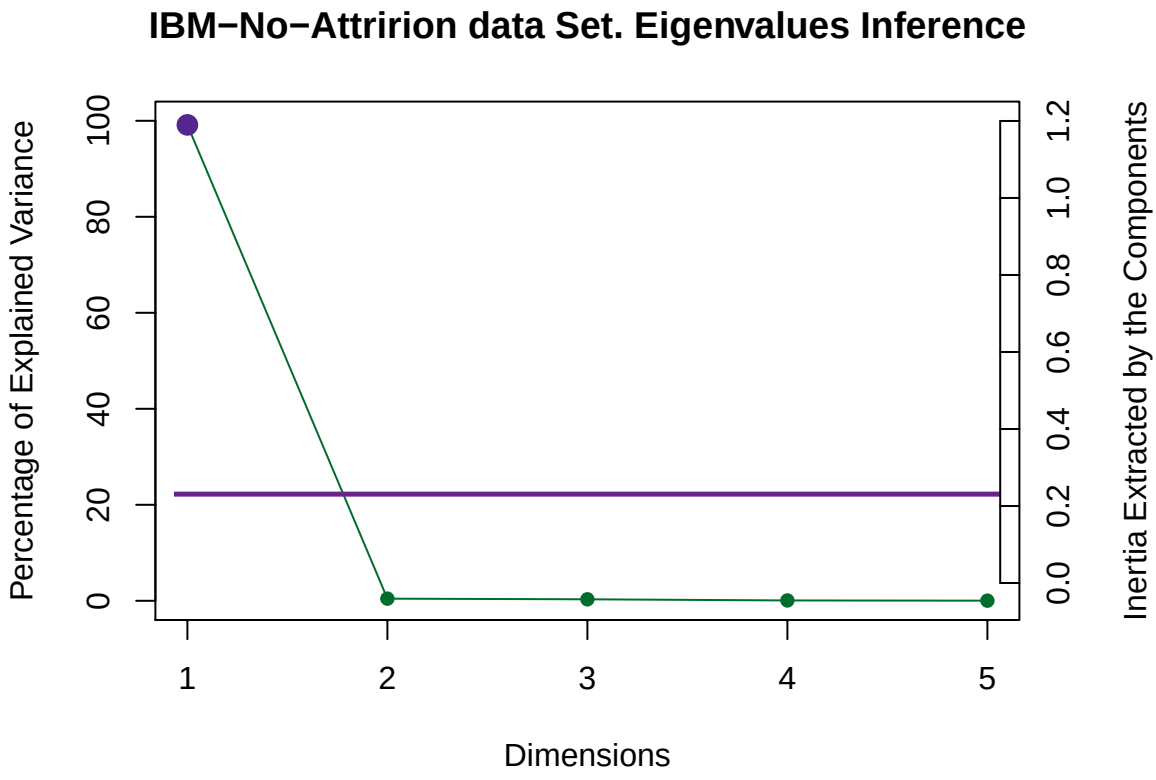
Scree Plot

A Scree Plot is a simple line segment plot that shows the fraction of total variance in the data as explained or represented by each PC.`(In the PCA literature, the plot is called a 'Scree' Plot because it often looks like a 'scree' slope, where rocks have fallen down and accumulated on the side of a mountain.)`The scree plot shows the eigenvalues, the amount of information on each component. The number of components (the dimensionality of the factor space) is min(nrow(DATA), ncol(DATA)) minus 1.

```
library(TExPosition)
library(data4PCCAR)
resPLSC <- tepPLS(data1,data2,DESIGN = design1,graphs = FALSE)
```

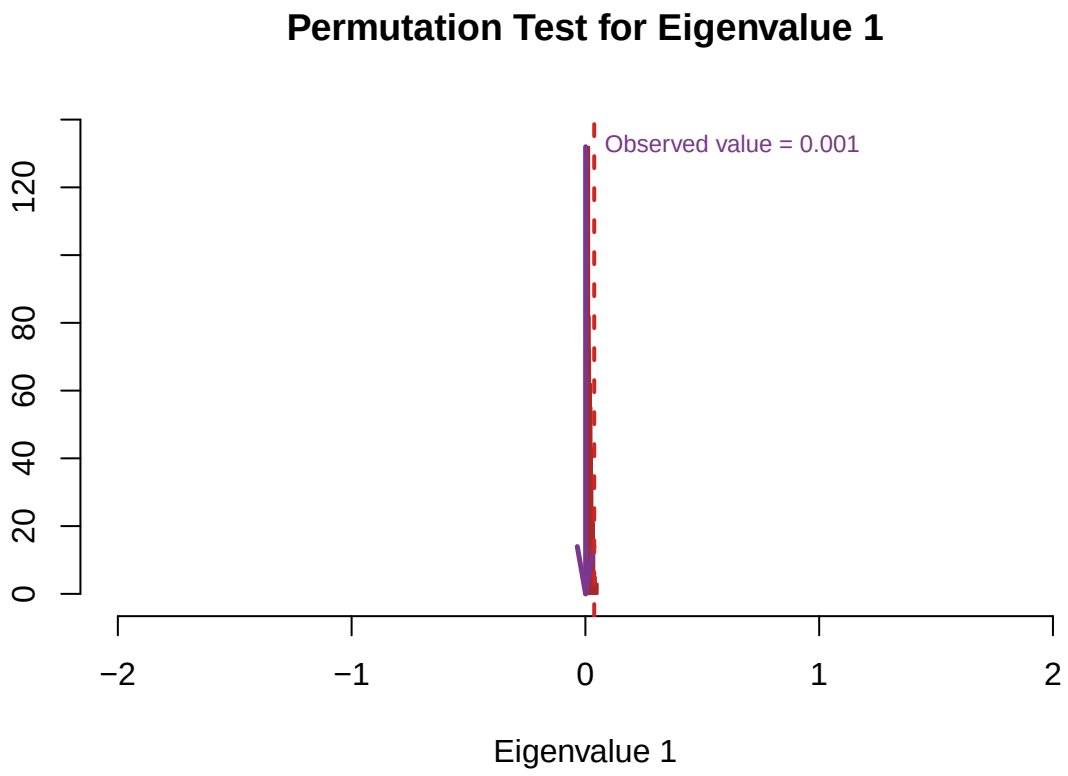
```
resPerm4PLSC <- perm4PLSC(data1, # First Data matrix
                          data2, # Second Data matrix
                          nIter = 1000 # How mny iterations
                          )
```

```
PlotScree(ev = resPLSC$TExPosition.Data$eigs,
          p.ev = resPerm4PLSC$pEigenvalues,
          title = 'IBM-No-Attririon data Set. Eigenvalues Inference',
          plotKaiser = TRUE
)
```



Permutation test for eigen-values

```
zeDim = 1
pH1 <- prettyHist(
  distribution = resPerm4PLSC$permEigenvalues[,zeDim],
           observed = resPerm4PLSC$pEigenvalues[zeDim],
           xlim = c(-2, 2), # needs to be set by hand
           breaks = 20,
           border = "brown",
           main = paste0("Permutation Test for Eigenvalue ",zeDim),
           xlab = paste0("Eigenvalue ",zeDim),
           ylab = "",
           counts = FALSE,
           cutoffs = c( 0.975))
```




```
b005.PermTest <- recordPlot()
```

5.1 Factor Maps for latent Variables

lx vs ly for Gender type (1st Component)

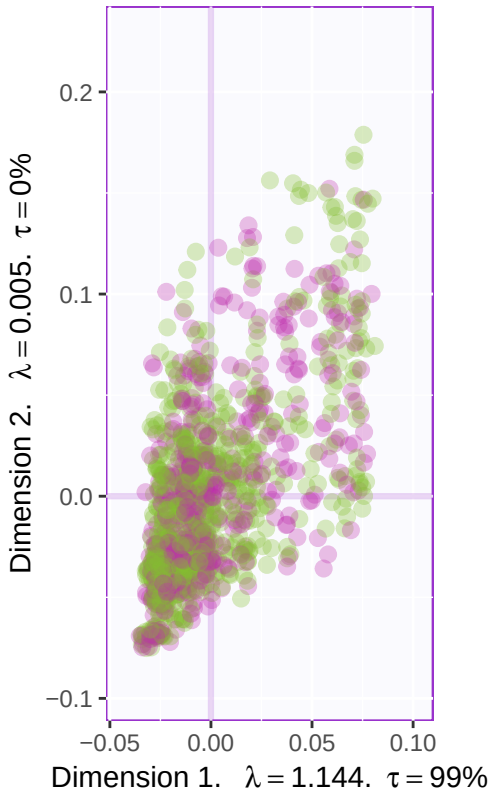
```
library(data4PCCAR)
library(PTCA4CATA)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(ggplot2)
library(ExPosition)
library(InPosition)
constraints1 <- resPLSC$Plotting.Data$constraints

df <- (cbind(resPLSC$TExPosition.Data$lx[,1],resPLSC$TExPosition.Data$ly[,1]))
```

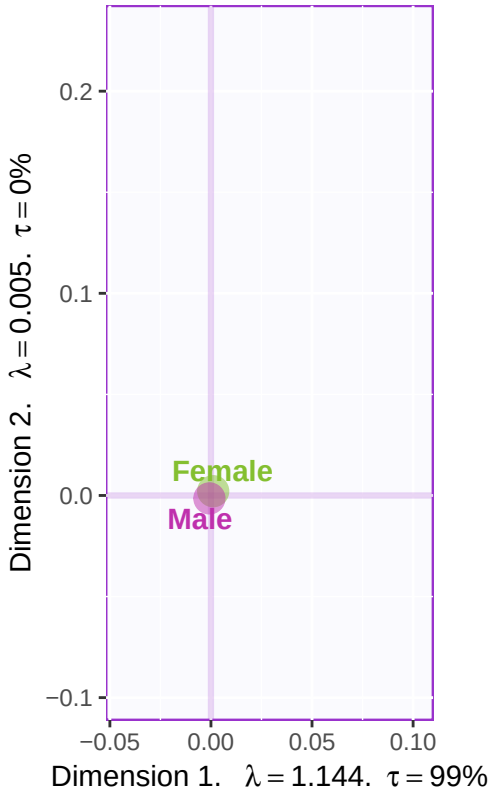
```
baseMap.i0 <- PTCA4CATA::createFactorMap(df,
                                         col.points    = resPLSC$Plotting.Data$fii.col,
                                         alpha.points =  .3)
label4Map0 <- createxyLabels.gen(1,2,
                                 lambda = resPLSC$TExPosition.Data$eigs,
                                 tau = resPLSC$TExPosition.Data$t)

# Plain map with color for the I-set
```

```r
aggMap.i0 <- baseMap.i0$zeMap_background + baseMap.i0$zeMap_dots + label4Map0
#-----------------------------------------------------------------------
print(aggMap.i0)
```

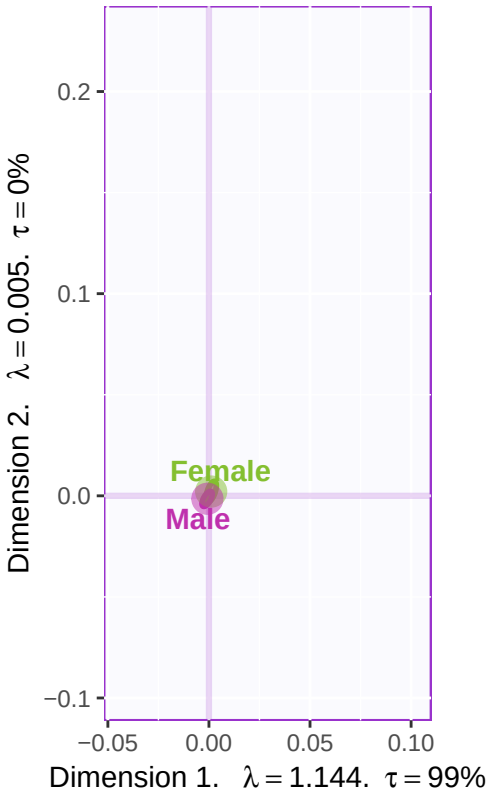



```r
col4data0 <- resPLSC$Plotting.Data$fii.col
col4Means0 <- unique(col4data0)
dataMeans0 <- getMeans(df, design1)

MapGroup0    <- PTCA4CATA::createFactorMap(dataMeans0,
                                axis1 = 1, axis2 = 2,
                                #constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means0,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means0,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                              col.background = adjustcolor("lavender",
                                                           alpha.f = 0.2),
                              force = 1, segment.size = 0)
```

```
aggMap.i.withMeans0 <- baseMap.i0$zeMap_background  +
  MapGroup0$zeMap_dots + MapGroup0$zeMap_text + label4Map0
#-----------------------------------------------------------------------

print(aggMap.i.withMeans0)
```

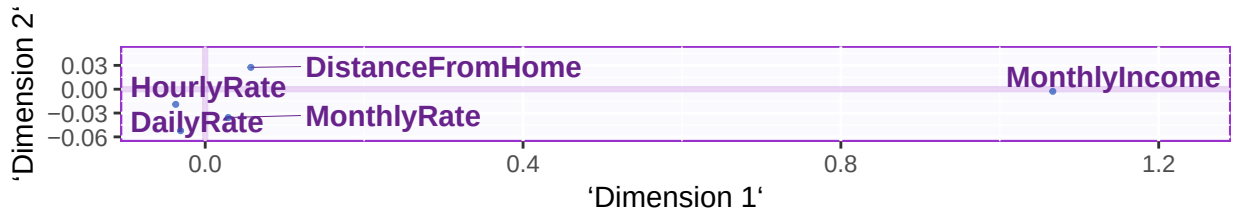



```
df <- as.matrix(df)
BootCube.Gr0 <- PTCA4CATA::Boot4Mean(df, design = design1,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli0 <- MakeCIEllipses(BootCube.Gr0$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means0,
                   p.level = .95
)


aggMap.i.withCI0 <-  baseMap.i0$zeMap_background + GraphElli0 + label4Map0 + MapGroup0$zeMap_text + Map
print(aggMap.i.withCI0)
```

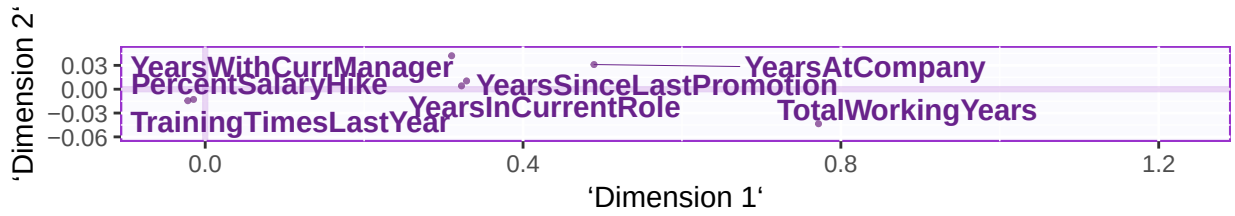


5.2 Factor Maps for J

```
constraints <- minmaxHelper(mat1 = resPLSC$TExPosition.Data$fi, mat2  = resPLSC$TExPosition.Data$fj)
baseMap.j2 <- createFactorMap(resPLSC$TExPosition.Data$fi,constraints = constraints,
                              col.points = resPLSC$Plotting.Data$fi.col, axis1 = 1, axis2 = 2,
                              cex = 1, pch = 20,
                              display.labels = TRUE
)
a2 <- baseMap.j2$zeMap + baseMap.j2$zeMap_dots
print(a2)
```




```
baseMap.j3 <- createFactorMap(resPLSC$TExPosition.Data$fj,constraints = constraints,
                              col.points = resPLSC$Plotting.Data$fj.col, axis1 = 1, axis2 = 2,
                              cex = 1, pch = 20,
                              display.labels = TRUE
)
a1 <- baseMap.j3$zeMap + baseMap.j3$zeMap_dots
print(a1)
```

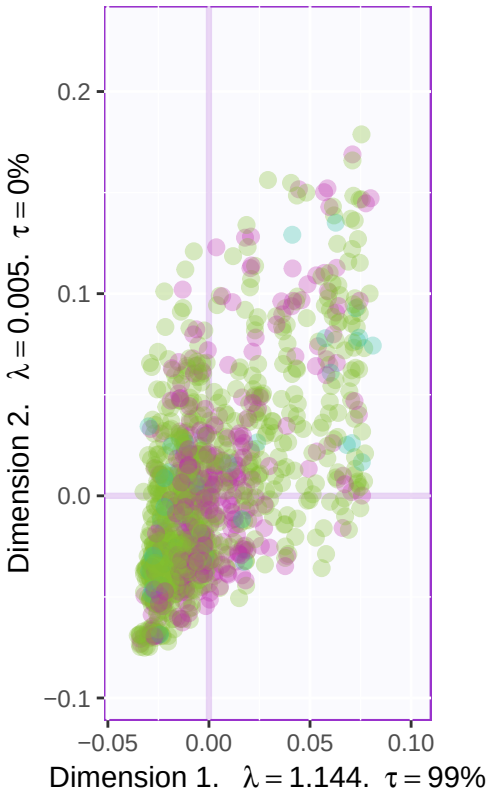


Departmenttype

```
resPLSC1 <- tepPLS(data1,data2,DESIGN = design2,graphs = FALSE)
```

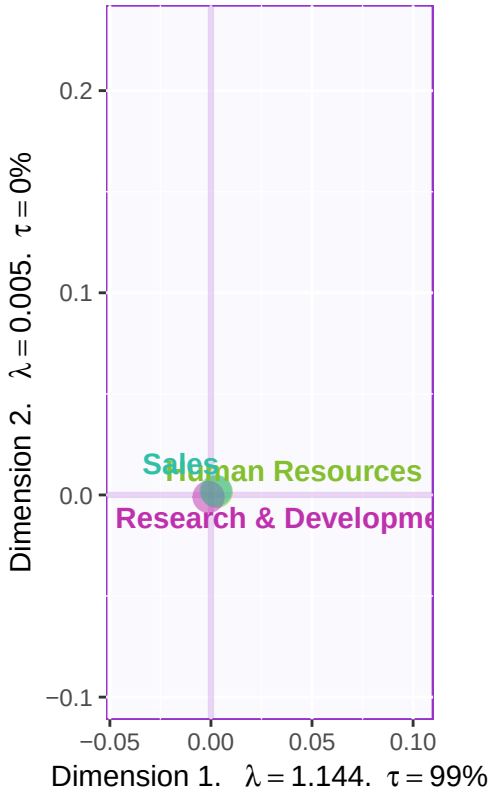
```
df2 <- (cbind(resPLSC1$TExPosition.Data$lx[,1],resPLSC1$TExPosition.Data$ly[,1]))
```

```
baseMap.i1 <- PTCA4CATA::createFactorMap(df2,
                                         col.points   = resPLSC1$Plotting.Data$fii.col,
                                         alpha.points =  .3)
label4Map <- createxyLabels.gen(1,2,
                                lambda = resPLSC1$TExPosition.Data$eigs,
                                tau = resPLSC1$TExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i1 <- baseMap.i1$zeMap_background + baseMap.i1$zeMap_dots + label4Map
#-----------------------------------------------------------------------
print(aggMap.i1)
```

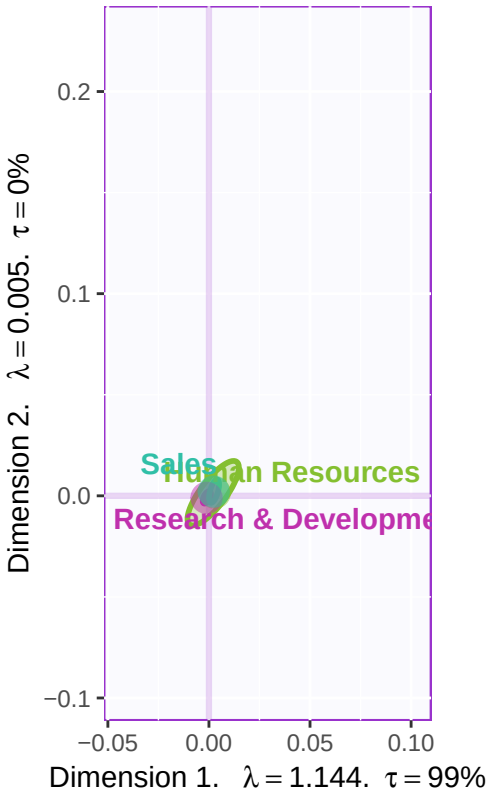



```
col4data1 <- resPLSC1$Plotting.Data$fii.col
col4Means1 <- unique(col4data1)
dataMeans1 <- getMeans(df2, design2)

MapGroup1    <- PTCA4CATA::createFactorMap(dataMeans1,
                                  axis1 = 1, axis2 = 2,
                                  #constraints = baseMap.i1$constraints,
                                  title = NULL,
                                  col.points = col4Means1,
                                  display.points = TRUE,
                                  pch = 19, cex = 5,
                                  display.labels = TRUE,
                                  col.labels = col4Means1,
                                  text.cex = 4,
                                  font.face = "bold",
                                  font.family = "sans",
                                  col.axes = "darkorchid",
                                  alpha.axes = 0.2,
                                  width.axes = 1.1,
                               col.background = adjustcolor("lavender",
                                                            alpha.f = 0.2),
                               force = 1, segment.size = 0)

aggMap.i.withMeans1 <- baseMap.i1$zeMap_background  +
  MapGroup1$zeMap_dots + MapGroup1$zeMap_text + label4Map
#-------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans1)
```




```
BootCube.Gr1 <- PTCA4CATA::Boot4Mean(df2, design = design2,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli1 <- MakeCIEllipses(BootCube.Gr1$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means1,
                   p.level = .95
)


aggMap.i.withCI1 <- baseMap.i1$zeMap_background +  GraphElli1 + MapGroup1$zeMap_text + MapGroup1$zeMap_

print(aggMap.i.withCI1)
```

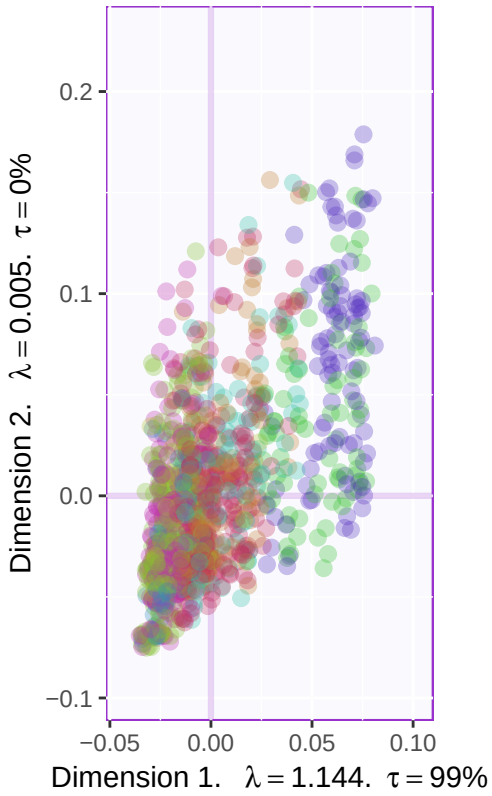


JobRole

```
resPLSC2 <- tepPLS(data1,data2,DESIGN = design3,graphs = FALSE)
```

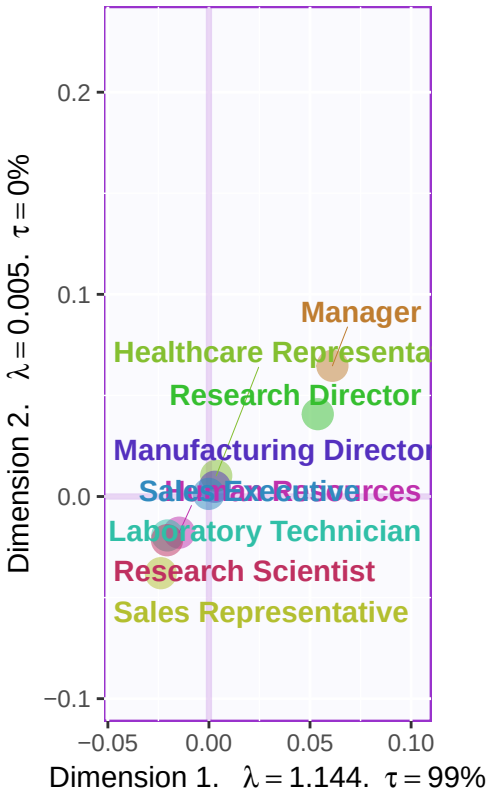
```
df4 <- (cbind(resPLSC2$TExPosition.Data$lx[,1],resPLSC2$TExPosition.Data$ly[,1]))
```

```
baseMap.i12 <- PTCA4CATA::createFactorMap(df4,
                                          col.points   = resPLSC2$Plotting.Data$fii.col,
                                          alpha.points =  .3)
label4Map2 <- createxyLabels.gen(1,2,
                                 lambda = resPLSC2$TExPosition.Data$eigs,
                                 tau = resPLSC2$TExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i12 <- baseMap.i12$zeMap_background + baseMap.i12$zeMap_dots + label4Map2
#-----------------------------------------------------------------------
print(aggMap.i12)
```

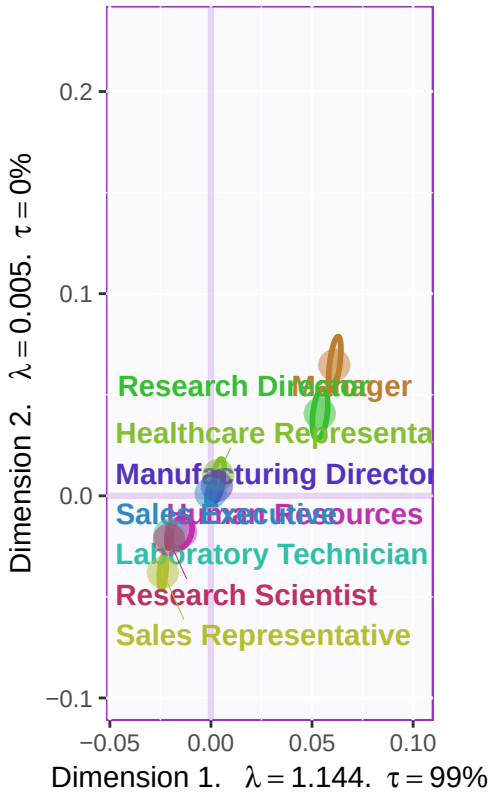



```
col4data2 <- resPLSC2$Plotting.Data$fii.col
col4Means2 <- unique(col4data2)
dataMeans2 <- getMeans(df4, design3)

MapGroup2    <- PTCA4CATA::createFactorMap(dataMeans2,
                                  axis1 = 1, axis2 = 2,
                                  #constraints = baseMap.i1$constraints,
                                  title = NULL,
                                  col.points = col4Means2,
                                  display.points = TRUE,
                                  pch = 19, cex = 5,
                                  display.labels = TRUE,
                                  col.labels = col4Means2,
                                  text.cex = 4,
                                  font.face = "bold",
                                  font.family = "sans",
                                  col.axes = "darkorchid",
                                  alpha.axes = 0.2,
                                  width.axes = 1.1,
                               col.background = adjustcolor("lavender",
                                                            alpha.f = 0.2),
                               force = 1, segment.size = 0)

aggMap.i.withMeans12 <- baseMap.i12$zeMap_background  +
  MapGroup2$zeMap_dots + MapGroup2$zeMap_text + label4Map2
#-------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans12)
```




```
BootCube.Gr2 <- PTCA4CATA::Boot4Mean(df4, design = design3,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli2 <- MakeCIEllipses(BootCube.Gr2$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means2,
                   p.level = .95
)


aggMap.i.withCI2 <- baseMap.i12$zeMap_background +  GraphElli2 + MapGroup2$zeMap_text + MapGroup2$zeMap_

print(aggMap.i.withCI2)
```

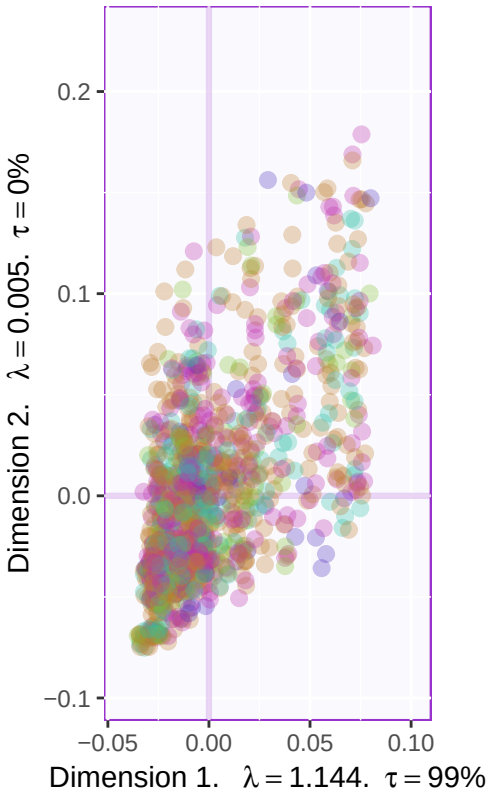


EducationField

```
resPLSC3 <- tepPLS(data1,data2,DESIGN = design4,graphs = FALSE)
```

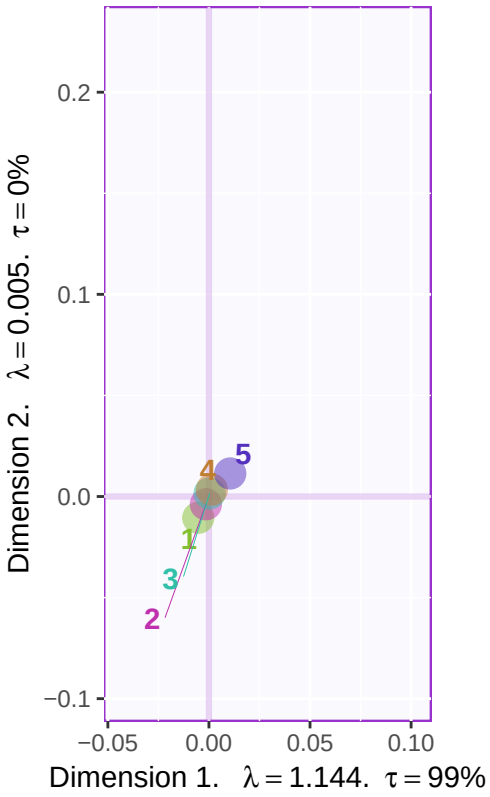
```
df6 <- (cbind(resPLSC3$TExPosition.Data$lx[,1],resPLSC3$TExPosition.Data$ly[,1]))
```

```
baseMap.i4 <- PTCA4CATA::createFactorMap(df6,
                                         col.points   = resPLSC3$Plotting.Data$fii.col,
                                         alpha.points =  .3)
label4Map4 <- createxyLabels.gen(1,2,
                                 lambda = resPLSC3$TExPosition.Data$eigs,
                                 tau = resPLSC3$TExPosition.Data$t)

# Plain map with color for the I-set
aggMap.i4 <- baseMap.i4$zeMap_background + baseMap.i4$zeMap_dots + label4Map4
#-----------------------------------------------------------------------
print(aggMap.i4)
```

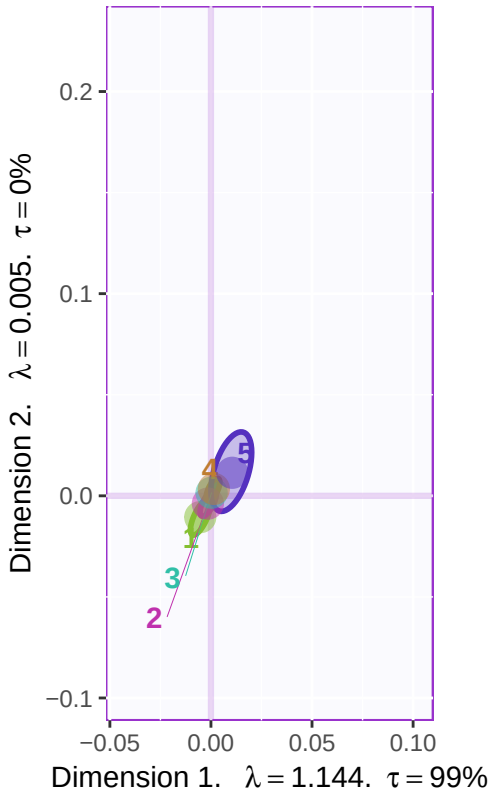



```
col4data4 <- resPLSC3$Plotting.Data$fii.col
col4Means4 <- unique(col4data4)
dataMeans4 <- getMeans(df6, design4)

MapGroup4    <- PTCA4CATA::createFactorMap(dataMeans4,
                                  axis1 = 1, axis2 = 2,
                                  #constraints = baseMap.i1$constraints,
                                  title = NULL,
                                  col.points = col4Means4,
                                  display.points = TRUE,
                                  pch = 19, cex = 5,
                                  display.labels = TRUE,
                                  col.labels = col4Means4,
                                  text.cex = 4,
                                  font.face = "bold",
                                  font.family = "sans",
                                  col.axes = "darkorchid",
                                  alpha.axes = 0.2,
                                  width.axes = 1.1,
                                col.background = adjustcolor("lavender",
                                                             alpha.f = 0.2),
                                force = 1, segment.size = 0)

aggMap.i.withMeans14 <- baseMap.i4$zeMap_background  +
  MapGroup4$zeMap_dots + MapGroup4$zeMap_text + label4Map4
#-------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans14)
```




```
BootCube.Gr4 <- PTCA4CATA::Boot4Mean(df6, design = design4,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli4 <- MakeCIEllipses(BootCube.Gr4$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means4,
                   p.level = .95
)

aggMap.i.withCI4 <- baseMap.i4$zeMap_background +  GraphElli4 + MapGroup4$zeMap_text + MapGroup4$zeMap_

print(aggMap.i.withCI4)
```

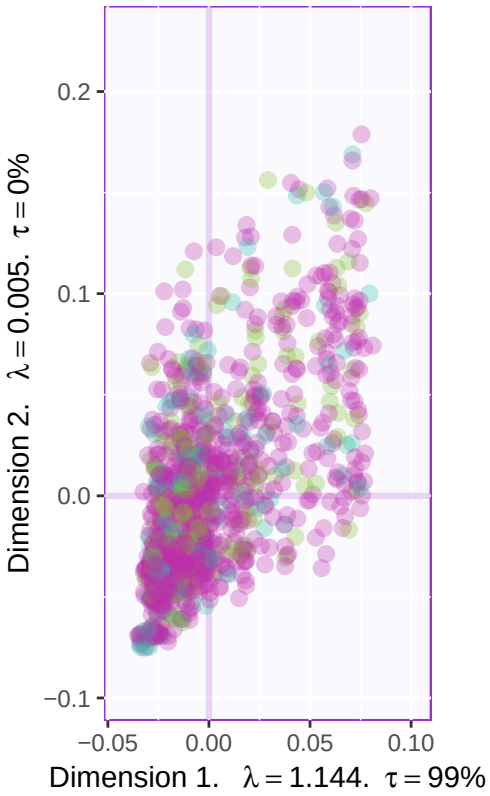


Business Travel

```
resPLSC4 <- tepPLS(data1,data2,DESIGN = design5,graphs = FALSE)
```

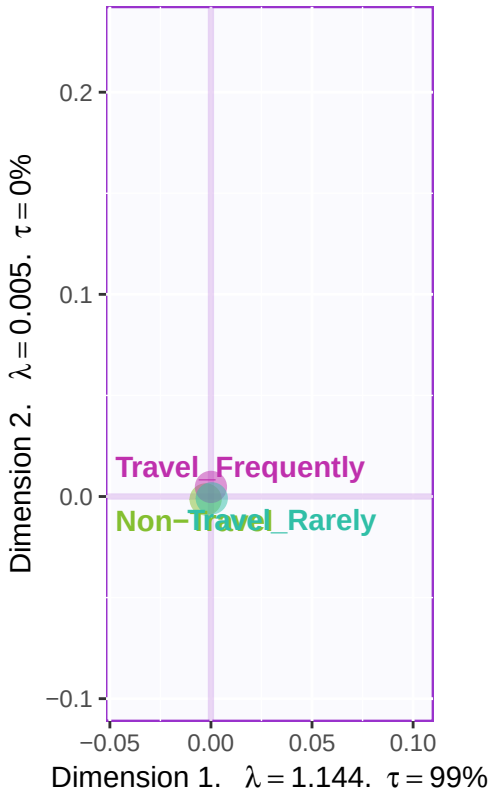
```
df7 <- (cbind(resPLSC4$TExPosition.Data$lx[,1],resPLSC4$TExPosition.Data$ly[,1]))
```

```
baseMap.i4 <- PTCA4CATA::createFactorMap(df7,
                                         col.points   = resPLSC4$Plotting.Data$fii.col,
                                         alpha.points =  .3)
label4Map4 <- createxyLabels.gen(1,2,
                                 lambda = resPLSC4$TExPosition.Data$eigs,
                                 tau = resPLSC4$TExPosition.Data$t)

# Plain map with color for the I-set
aggMap.i4 <- baseMap.i4$zeMap_background + baseMap.i4$zeMap_dots + label4Map4
#-----------------------------------------------------------------------
print(aggMap.i4)
```

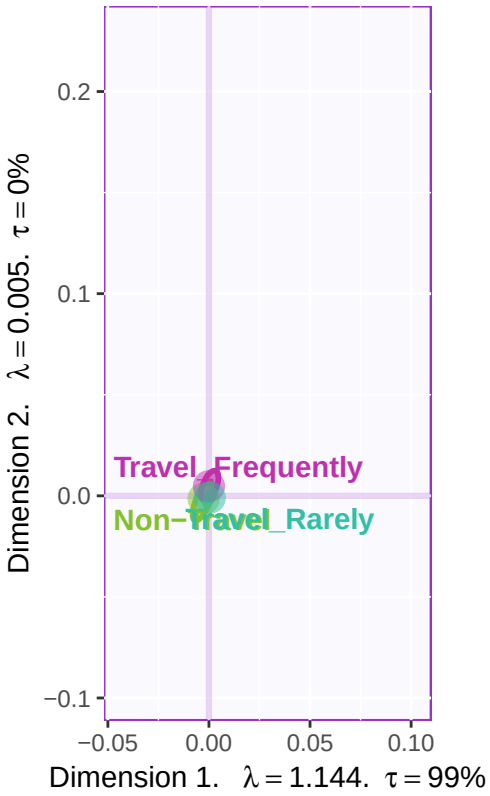



```
col4data4 <- resPLSC4$Plotting.Data$fii.col
col4Means4 <- unique(col4data4)
dataMeans4 <- getMeans(df6, design5)

MapGroup4    <- PTCA4CATA::createFactorMap(dataMeans4,
                                  axis1 = 1, axis2 = 2,
                                  #constraints = baseMap.i1$constraints,
                                  title = NULL,
                                  col.points = col4Means4,
                                  display.points = TRUE,
                                  pch = 19, cex = 5,
                                  display.labels = TRUE,
                                  col.labels = col4Means4,
                                  text.cex = 4,
                                  font.face = "bold",
                                  font.family = "sans",
                                  col.axes = "darkorchid",
                                  alpha.axes = 0.2,
                                  width.axes = 1.1,
                               col.background = adjustcolor("lavender",
                                                            alpha.f = 0.2),
                               force = 1, segment.size = 0)

aggMap.i.withMeans14 <- baseMap.i4$zeMap_background  +
  MapGroup4$zeMap_dots + MapGroup4$zeMap_text + label4Map4
#-----------------------------------------------------------------------
```

```
print(aggMap.i.withMeans14)
```

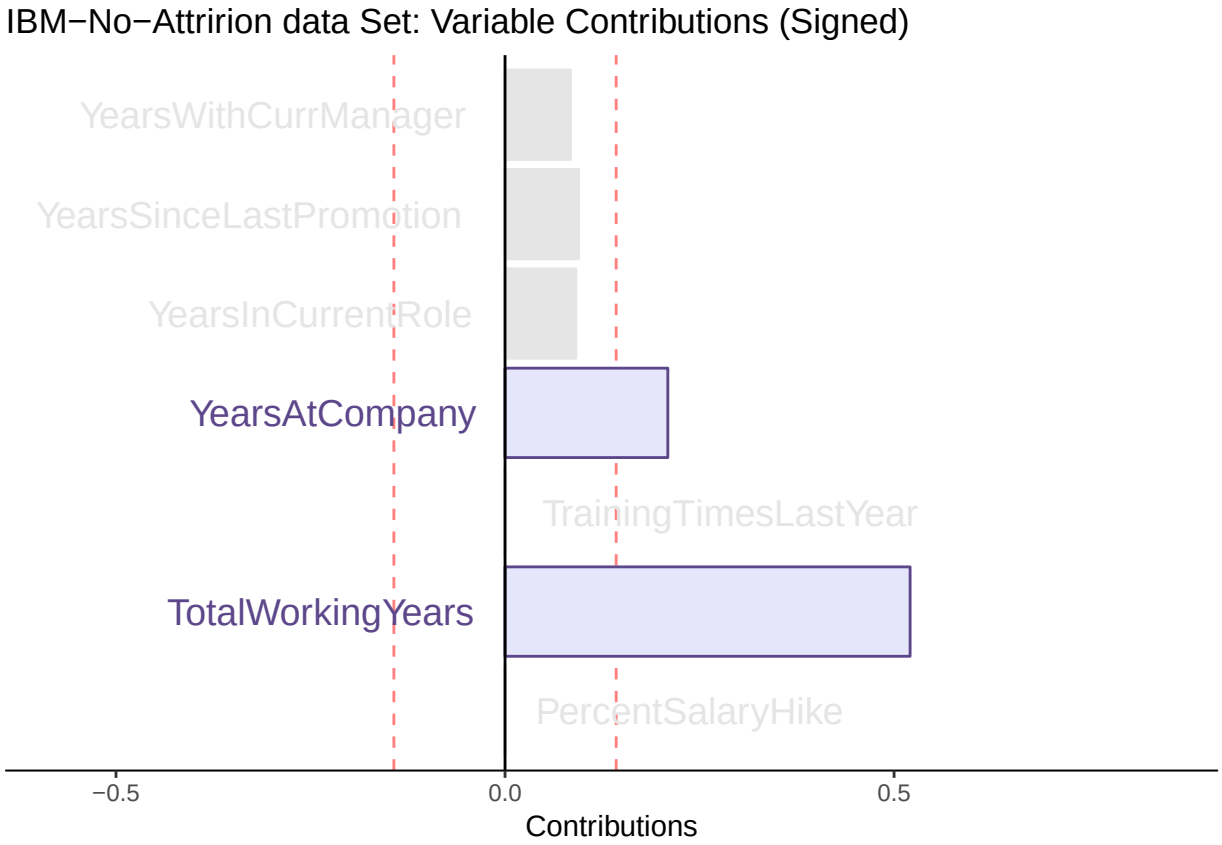



```
BootCube.Gr4 <- PTCA4CATA::Boot4Mean(df7, design = design5,
                                     niter = 100,
                                     suppressProgressBar = TRUE)

GraphElli4 <- MakeCIEllipses(BootCube.Gr4$BootCube[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means4,
                   p.level = .95
)

aggMap.i.withCI4 <- baseMap.i4$zeMap_background +  GraphElli4 + MapGroup4$zeMap_text + MapGroup4$zeMap_

print(aggMap.i.withCI4)
```

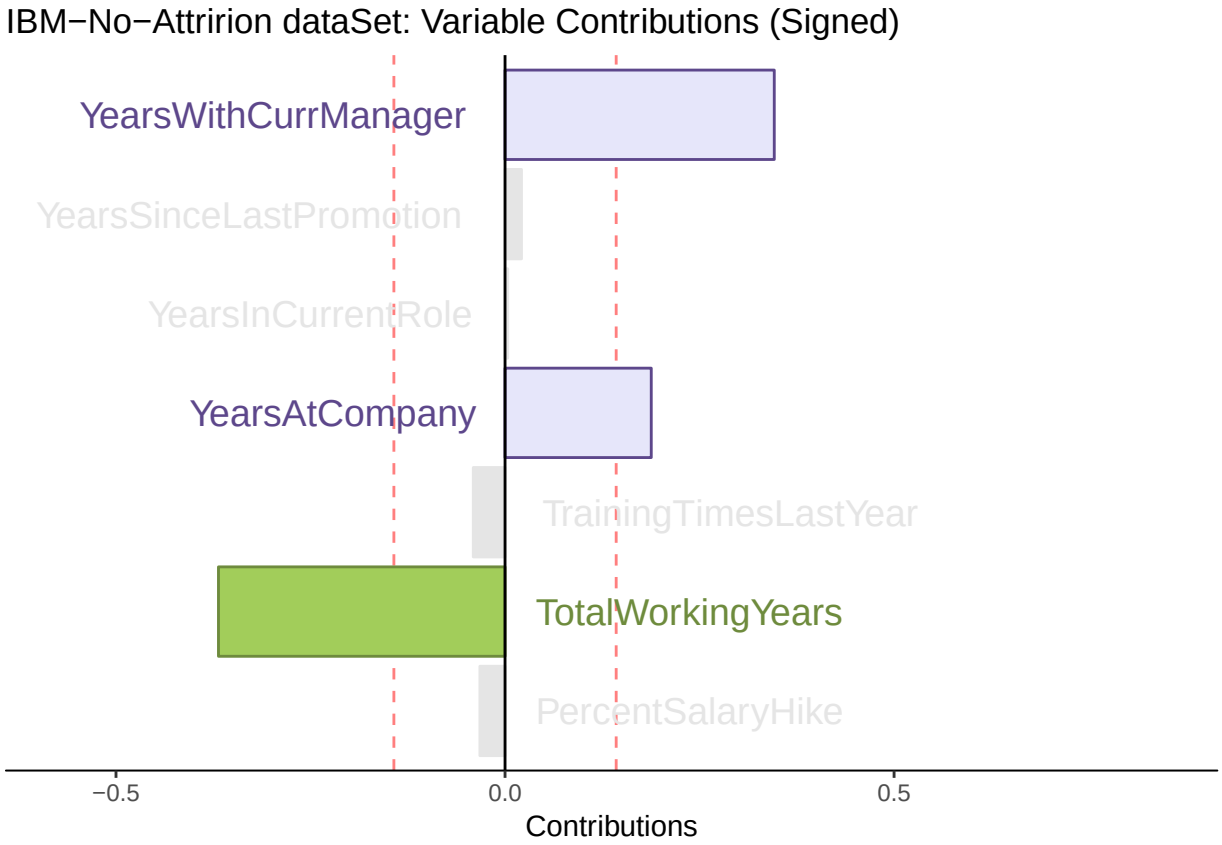


5.3 Contributions for Variables

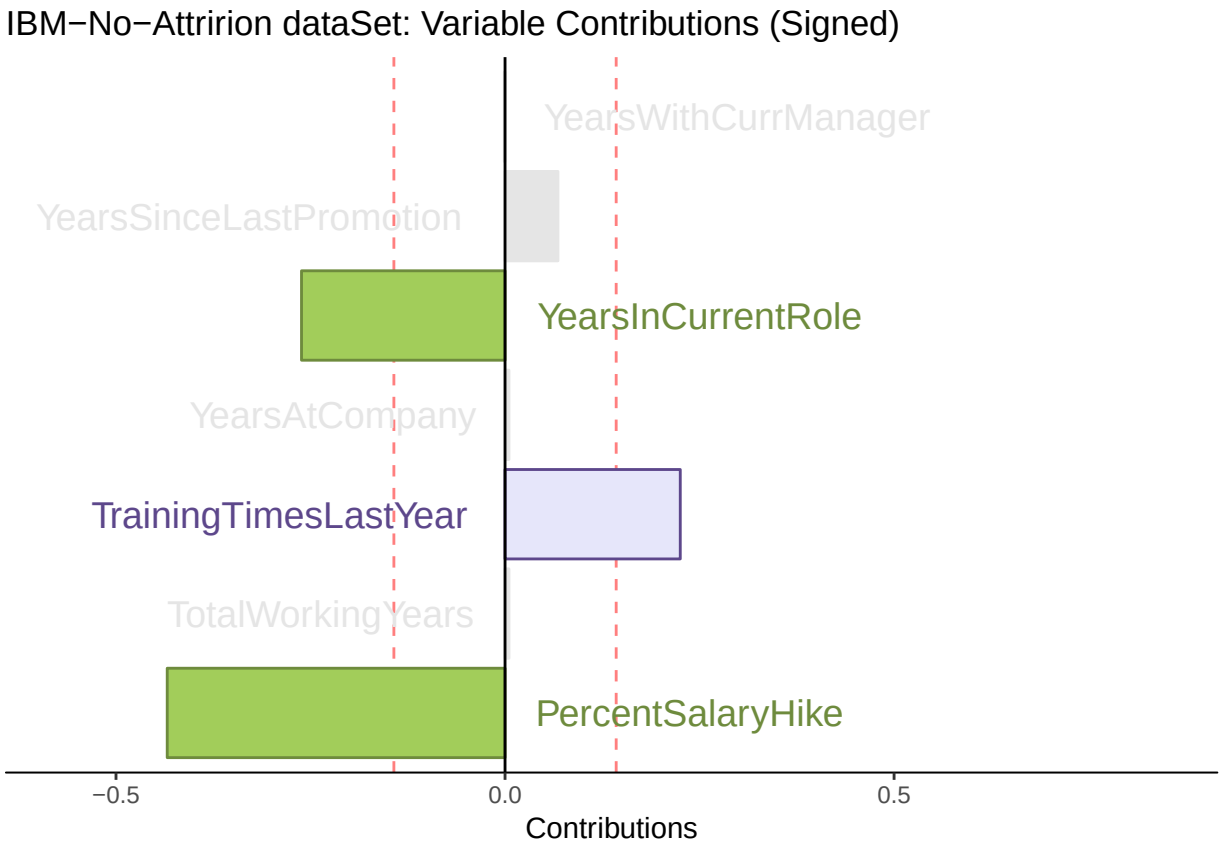
```
signed.ctrJ <- resPLSC$TExPosition.Data$cj * sign(resPLSC$TExPosition.Data$fj)
b003.ctrJ.s.1 <- PrettyBarPlot2(signed.ctrJ[,1],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                               # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'IBM-No-Attririon data Set: Variable Contributions (Signed)',
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ)), horizontal = FALSE
)
print(b003.ctrJ.s.1)
```




```
b004.ctrJ.s.2 <- PrettyBarPlot2(signed.ctrJ[,2],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                               # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'IBM-No-Attririon dataSet: Variable Contributions (Signed)',
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ)), horizontal = FALSE
)
print(b004.ctrJ.s.2)
```

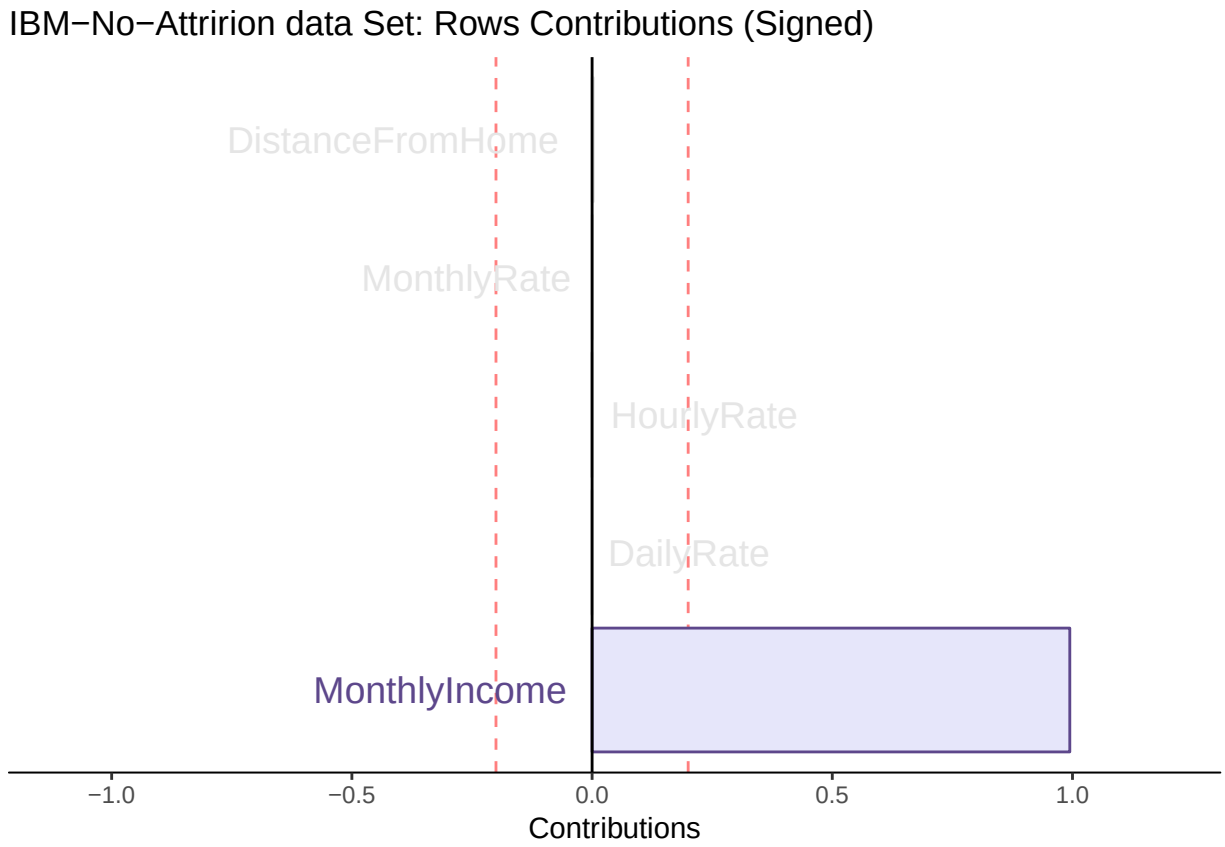



```
b004.ctrJ.s.3 <- PrettyBarPlot2(signed.ctrJ[,3],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                               # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'IBM-No-Attririon dataSet: Variable Contributions (Signed)',
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ)),horizontal = FALSE
)
print(b004.ctrJ.s.3)
```

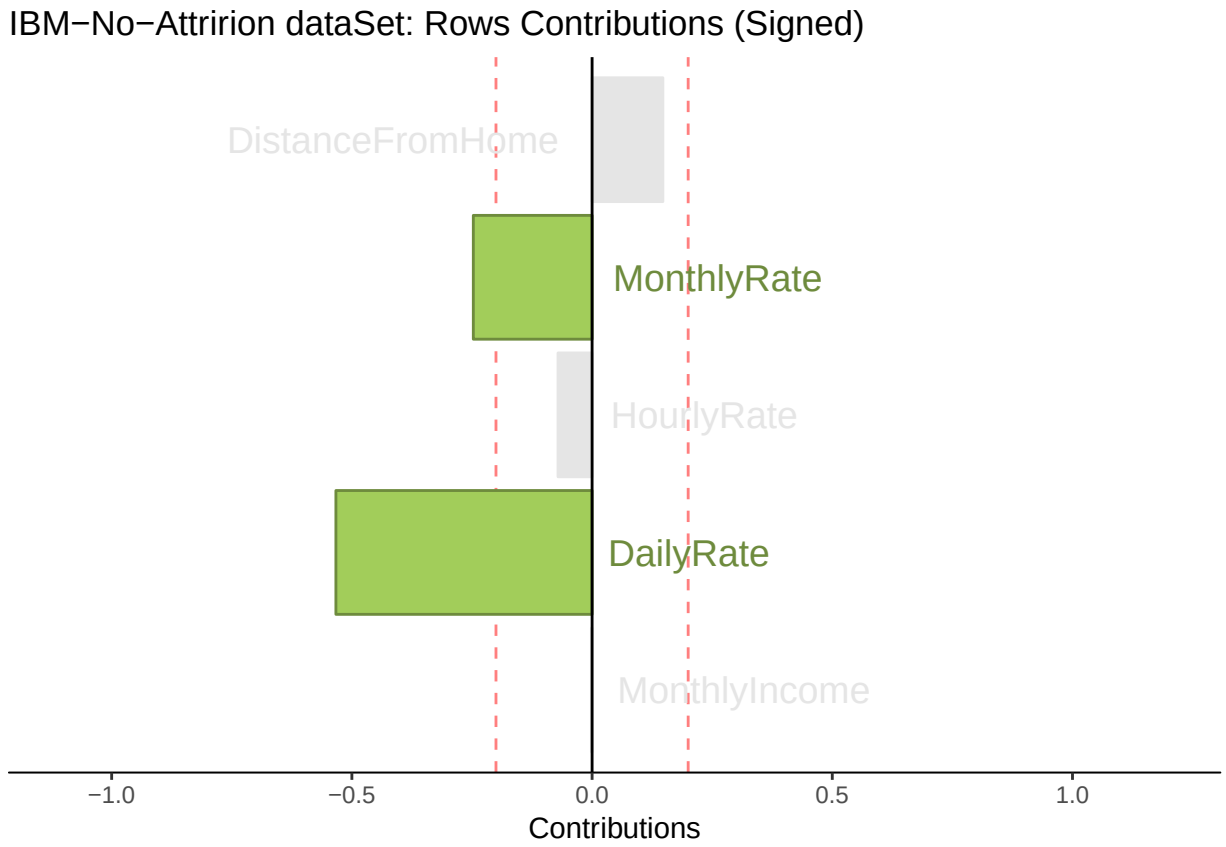


5.4 Contribution for Rows

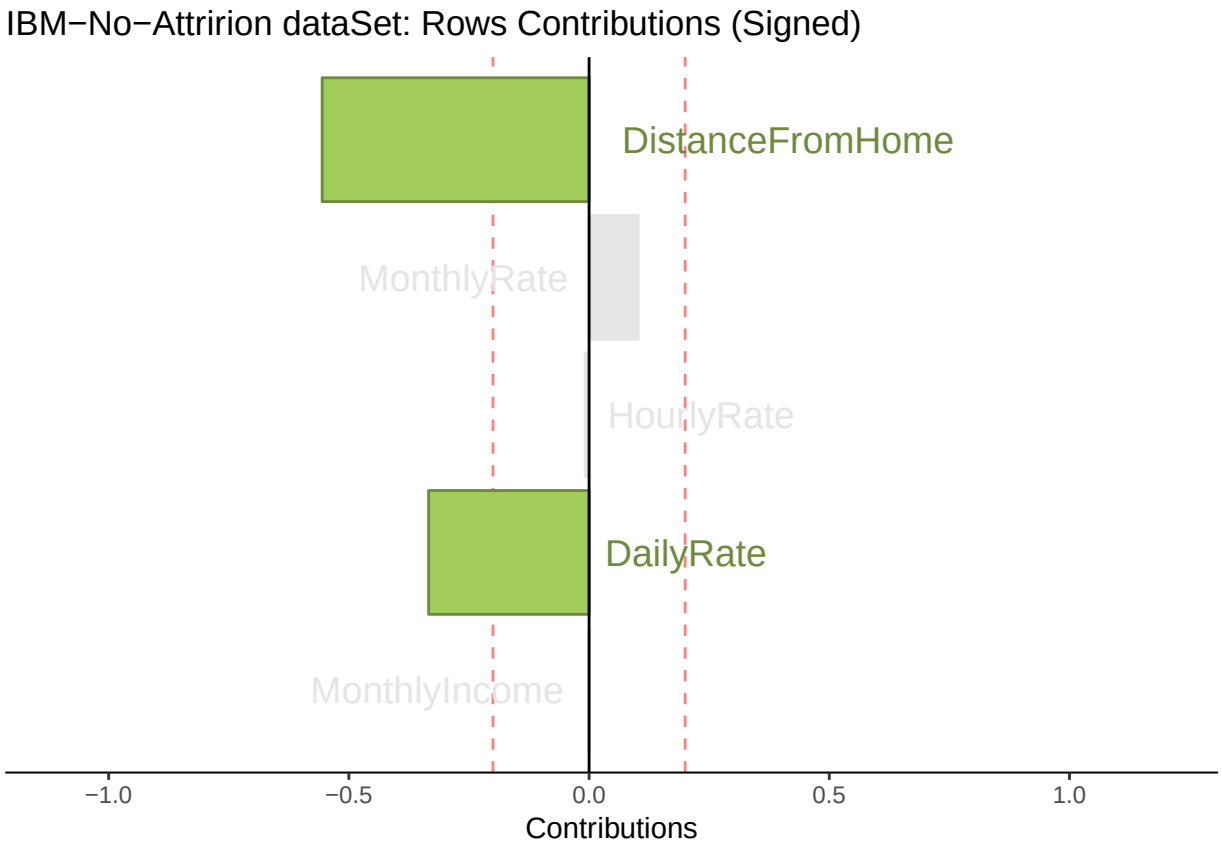
```
signed.ctri <- resPLSC$TExPosition.Data$ci * sign(resPLSC$TExPosition.Data$fi)
b003.ctri.s.1 <- PrettyBarPlot2(signed.ctri[,1],
                                threshold = 1 / NROW(signed.ctri),
                                font.size = 5,
                               # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'IBM-No-Attririon data Set: Rows Contributions (Signed)',
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctri), 1.2*max(signed.ctri)), horizontal = FALSE
)
print(b003.ctri.s.1)
```




```
b004.ctri.s.2 <- PrettyBarPlot2(signed.ctri[,2],
                                threshold = 1 / NROW(signed.ctri),
                                font.size = 5,
                               # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'IBM-No-Attririon dataSet: Rows Contributions (Signed)',
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctri), 1.2*max(signed.ctri)) , horizontal = FALS
)
print(b004.ctri.s.2)
```

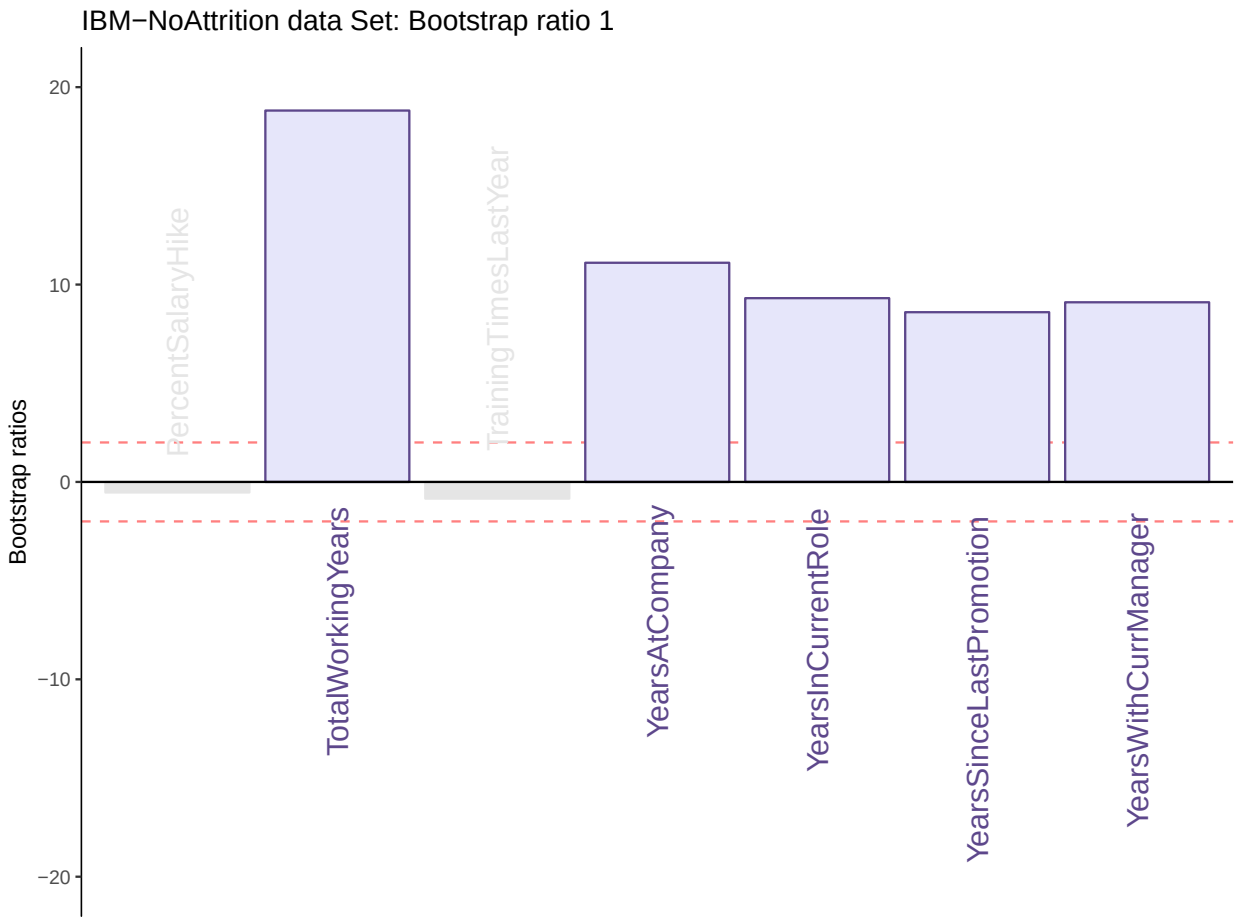



```
b004.ctri.s.3 <- PrettyBarPlot2(signed.ctri[,3],
                                threshold = 1 / NROW(signed.ctri),
                                font.size = 5,
                               # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'IBM-No-Attririon dataSet: Rows Contributions (Signed)',
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctri), 1.2*max(signed.ctri)),horizontal = FALSE
)
print(b004.ctri.s.3)
```

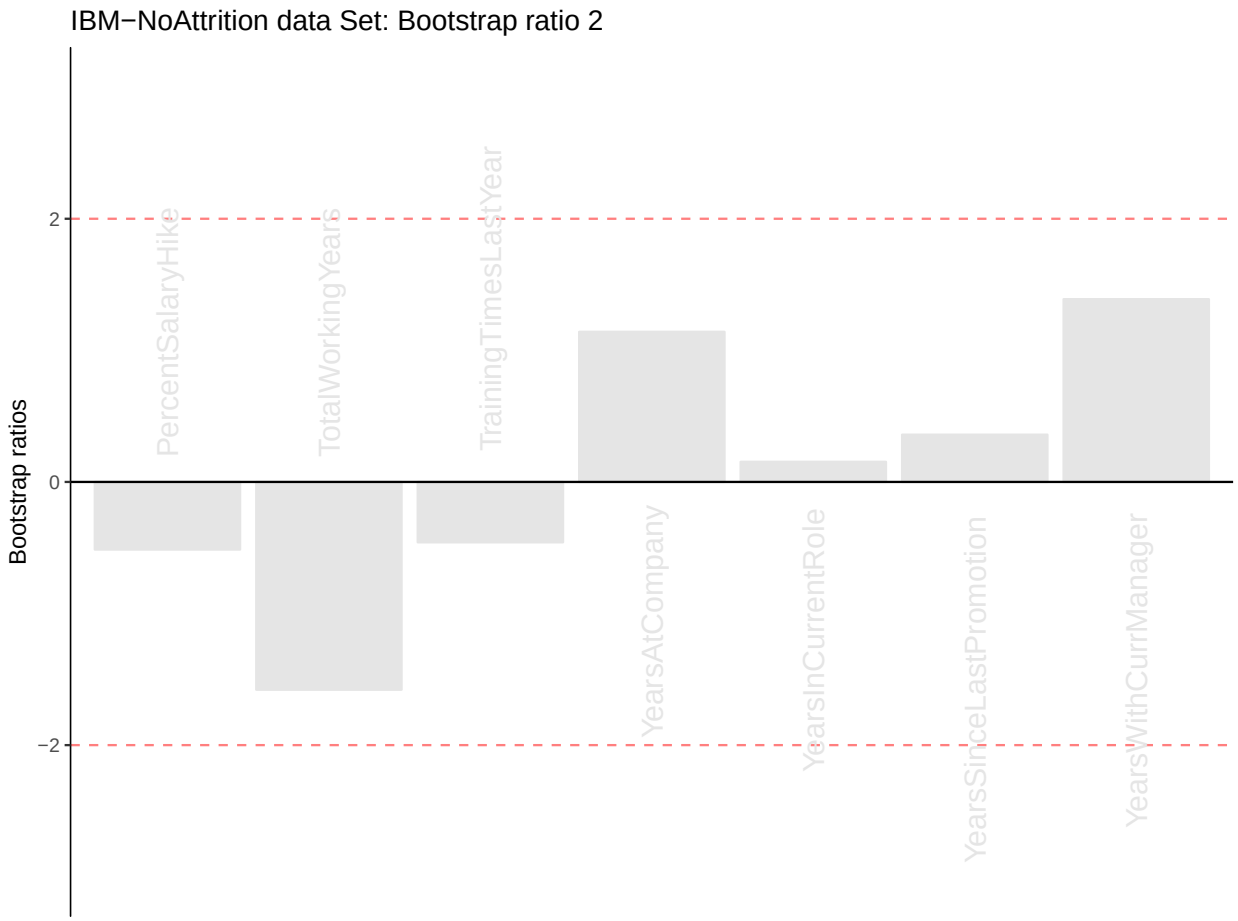


5.5 Bootstrap Ratios for Variables

```
resBoot4PLSC <- Boot4PLSC(data1, # First Data matrix
                          data2, # Second Data matrix
                          nIter = 1000, # How many iterations
                          Fi = resPLSC$TExPosition.Data$fi,
                          Fj = resPLSC$TExPosition.Data$fj,
                          nf2keep = 3,
                          critical.value = 2,
                          # To be implemented later
                          # has no effect currently
                          alphaLevel = .05)
print(resBoot4PLSC)
laDim = 1
ba001.BR1 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.j[,laDim],
                            threshold = 2,
                            font.size = 5,
                            #color4bar = gplots::col2hex(col4J.ibm),
                            main = paste0( 'IBM-NoAttrition data Set: Bootstrap ratio ',laDim),
                            ylab = 'Bootstrap ratios'
                            #ylim = c(1.2*min(BR[,laDim]), 1.2*max(BR[,laDim]))
)
print(ba001.BR1)
```



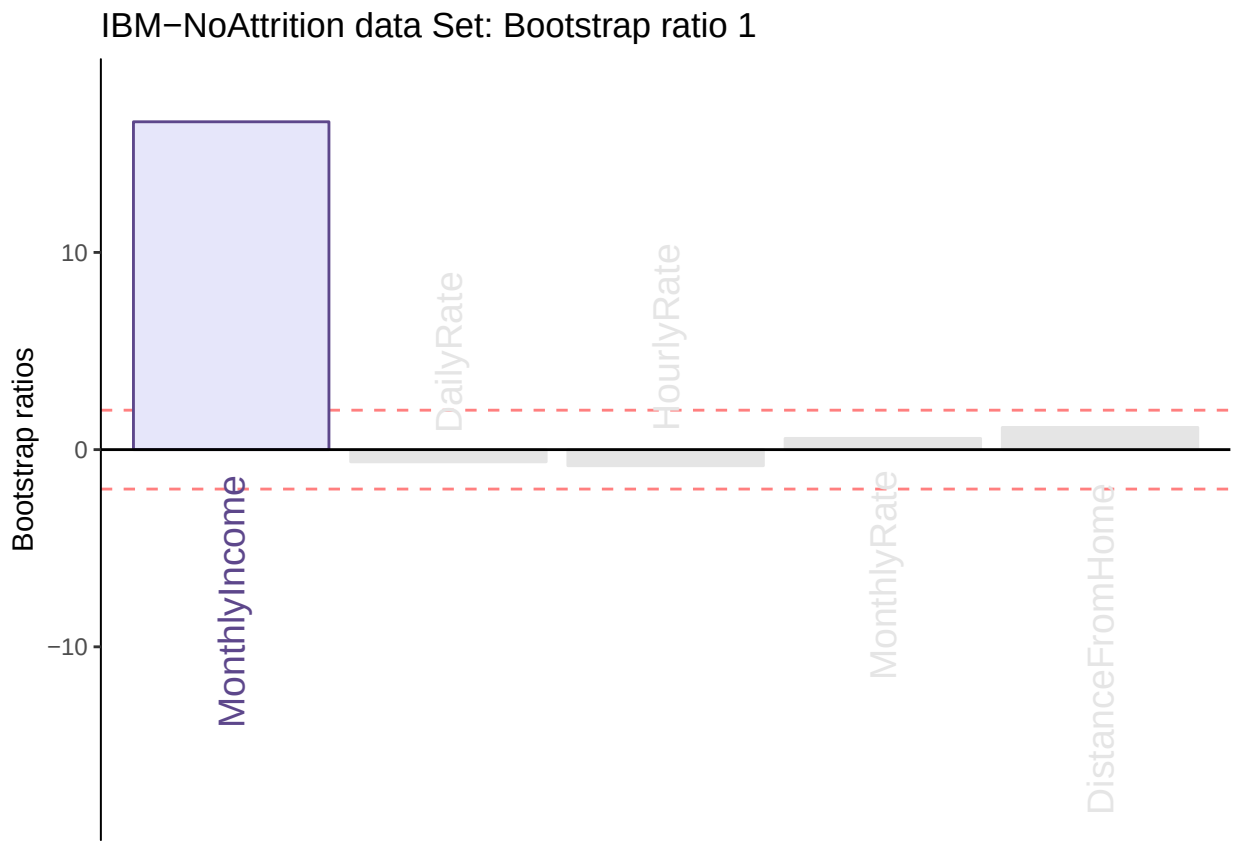

```
#
laDim = 2
ba002.BR2 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.j[,laDim],
                            threshold = 2,
                            font.size = 5,
                            #color4bar = gplots::col2hex(col4J.ibm),
                            main = paste0(
                              'IBM-NoAttrition data Set: Bootstrap ratio ',laDim),
                            ylab = 'Bootstrap ratios'
)
print(ba002.BR2)
```



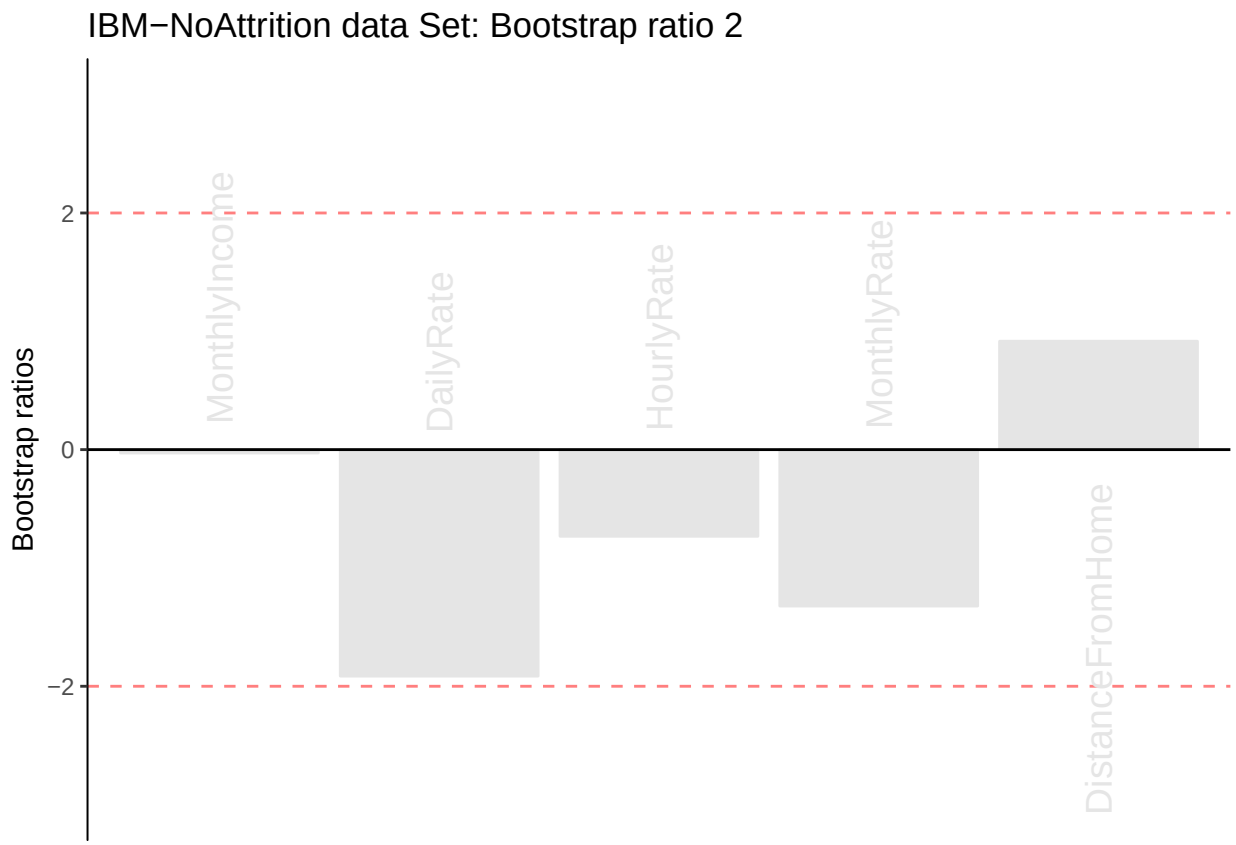

```
laDim = 3
ba002.BR3 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.j[,laDim],
                            threshold = 2,
                            font.size = 5,
                            main = paste0(
                              'IBM-NoAttrition data Set: Bootstrap ratio ',laDim),
                            ylab = 'Bootstrap ratios'
)
print(ba002.BR3)
```

5.6 Bootstrap Ratios for Rows

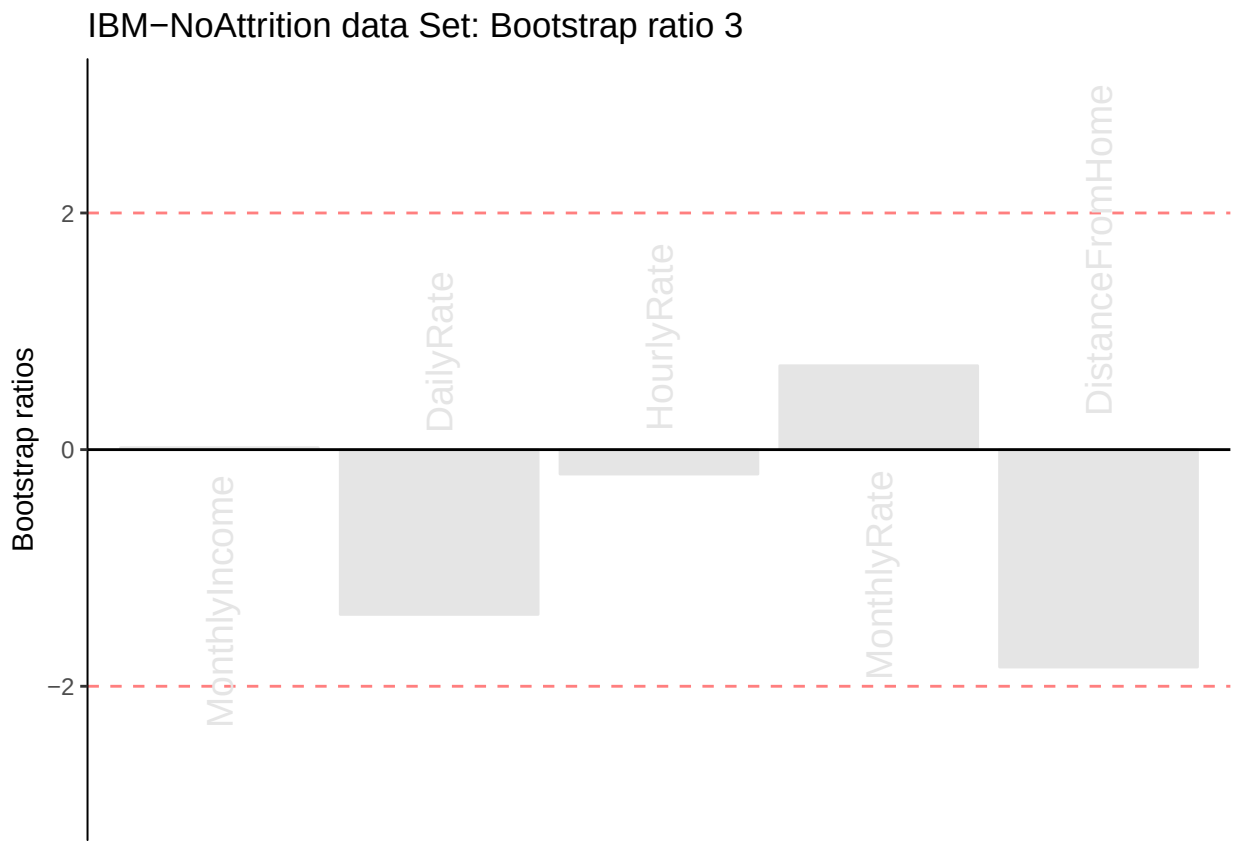
```
laDim = 1
ba001.BR11 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.i[,laDim],
                             threshold = 2,
                             font.size = 5,
                             #color4bar = gplots::col2hex(col4J.ibm),
                             main = paste0( 'IBM-NoAttrition data Set: Bootstrap ratio ',laDim),
                             ylab = 'Bootstrap ratios'
                             #ylim = c(1.2*min(BR[,laDim]), 1.2*max(BR[,laDim]))
)
print(ba001.BR11)
```




```
#
laDim = 2
ba002.BR21 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.i[,laDim],
                             threshold = 2,
                             font.size = 5,
                             #color4bar = gplots::col2hex(col4J.ibm),
                             main = paste0(
                               'IBM-NoAttrition data Set: Bootstrap ratio ',laDim),
                             ylab = 'Bootstrap ratios'
)
print(ba002.BR21)
```




```
laDim = 3
ba002.BR31 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.i[,laDim],
                             threshold = 2,
                             font.size = 5,
                             main = paste0(
                               'IBM-NoAttrition data Set: Bootstrap ratio ',laDim),
                             ylab = 'Bootstrap ratios'
)
print(ba002.BR31)
```



5.7 Summary

When we interpret the factor scores and loadings together, the PLS revealed the same results as the PCA:

- Usually the Research and Development people have more work-experience & Monthly Income in comparison to other department type
- Generally, Managers and Directors are the ones with PhD & Master's with higher job level, Sales representative and Lab Technician have no college education while Healthcare Representatives & Manufacturing Directors have Bachelor's degree
- People who are Managers and Research Directors have the highest pay and are more seasoned while people who are Sales representative and Lab Technician have low pay and less experience
- It is also observed that the Mangers and Research Directors travel the most, could be their meetings and conferences while the HR, Sales representative,Lab Technician, Research Scientists travel occasionally
- For the factor plot the gender type does not show any significant inference as the mean are almost overlapping each other and for the department type similar results can be seen as R&D, Sales and HR all intersect their mean confidence Intervals.

```
options(knitr.duplicate.label = 'allow')
```

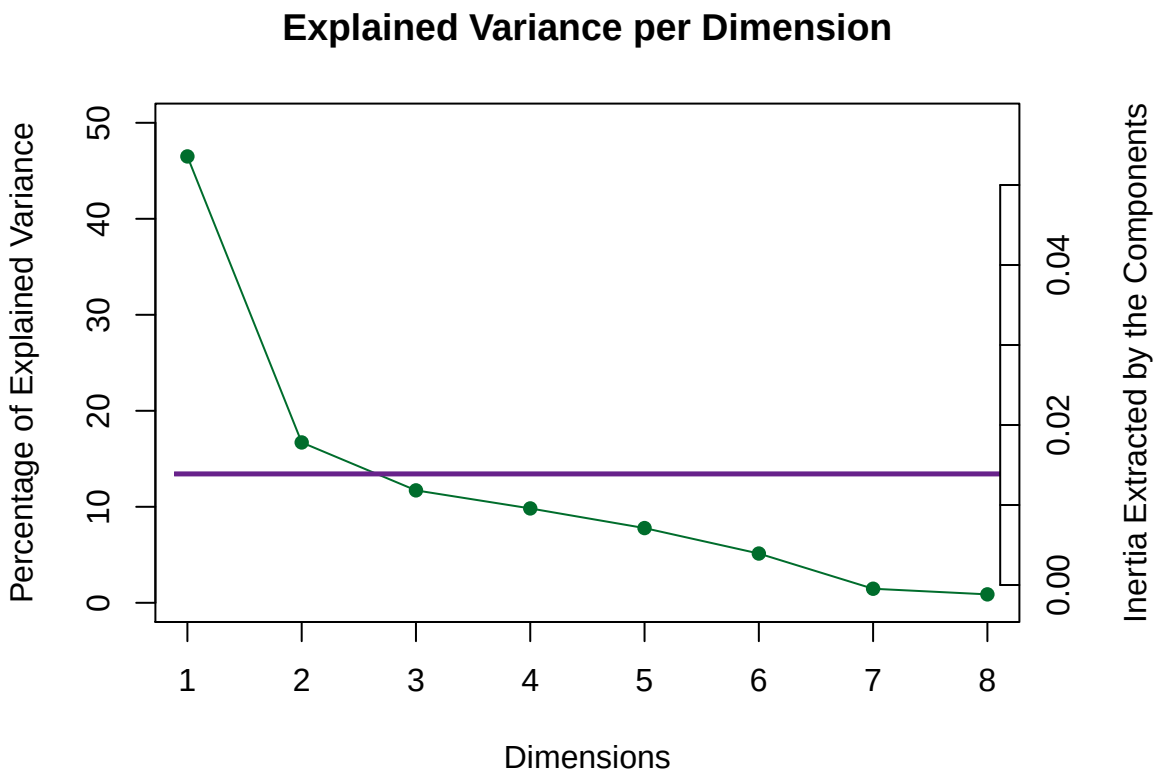
6 BADA

Barycentric discriminant analysis (BADA) is a robust version of discriminant analysis that is used to assign, to pre-defined groups (also called categories), observations described by multiple variables.The goal of BADA is to combine the measurements to create new variables (called components or discriminant variables) that best separate the categories. These discriminant variables are also used to assign the original observations or new observations to the a-priori defined categories. Barycentric discriminant analysis is a robust version of discriminant analysis that is used when multiple measurements describe a set of observations in which each observation belongs to one category (i.e., group) from a set of a priori defined categories. BADA combines the original variables to create new variables that best separate the groups and that can also be used to optimally assign old or new observations to these categories. The quality of the performance is evaluated by cross-validation techniques that estimate the performance of the classification model for new observations. BADA is a very versatile technique that can be declined in several different varieties that can handle, for example, qualitative data and data structured in blocks. This versatility make BADA particularly suited for the analysis of multi-modal and Big data.

```
datae <- data[,8]
datar <- data[,5]
resBADA <- tepBADA(DATA = data1,
                   scale = 'SS1', center = TRUE,
                   DESIGN = datae,
                   make_design_nominal = TRUE,
                   group.masses = NULL,
                   weights = NULL, graphs =  FALSE)

resBADA1 <- tepBADA(DATA = data1,
                   scale = 'SS1', center = TRUE,
                   DESIGN = datar,
                   make_design_nominal = TRUE,
                   group.masses = NULL,
                   weights = NULL, graphs =  FALSE)
```

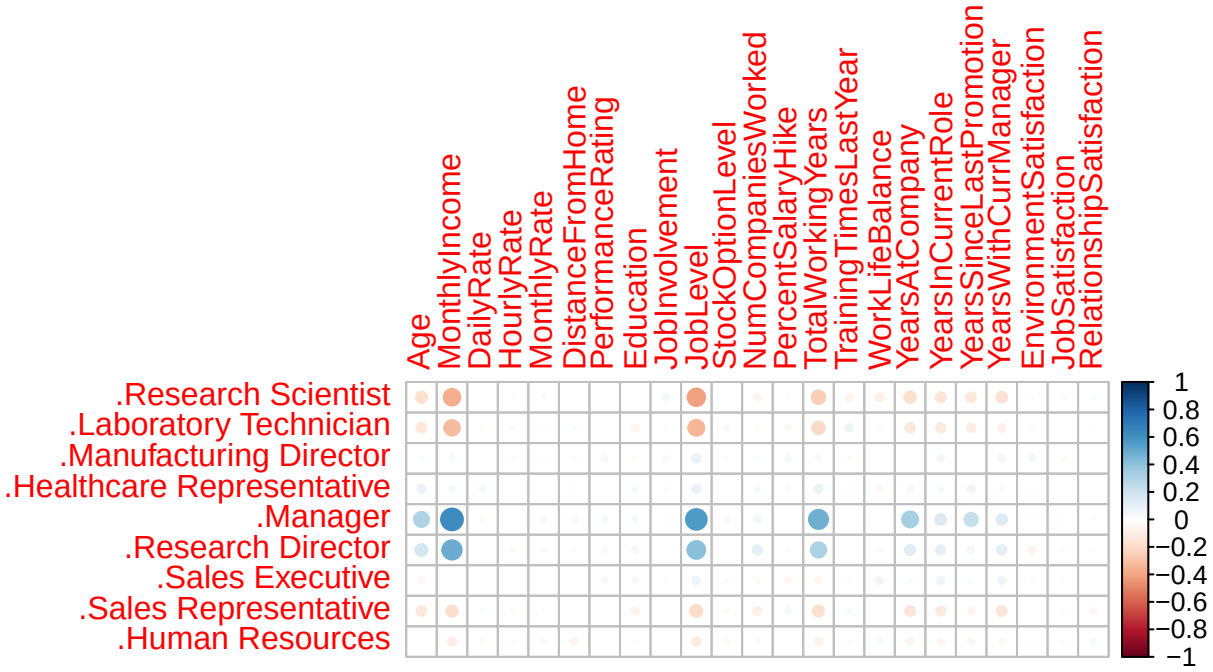
ScreePlot

```
PlotScree(ev = resBADA$TExPosition.Data$eigs,
          p.ev = NULL, max.ev = NULL, alpha = 0.05,
          col.ns = "#006D2C", col.sig = "#54278F",
          title = "Explained Variance per Dimension",plotKaiser = TRUE)
```

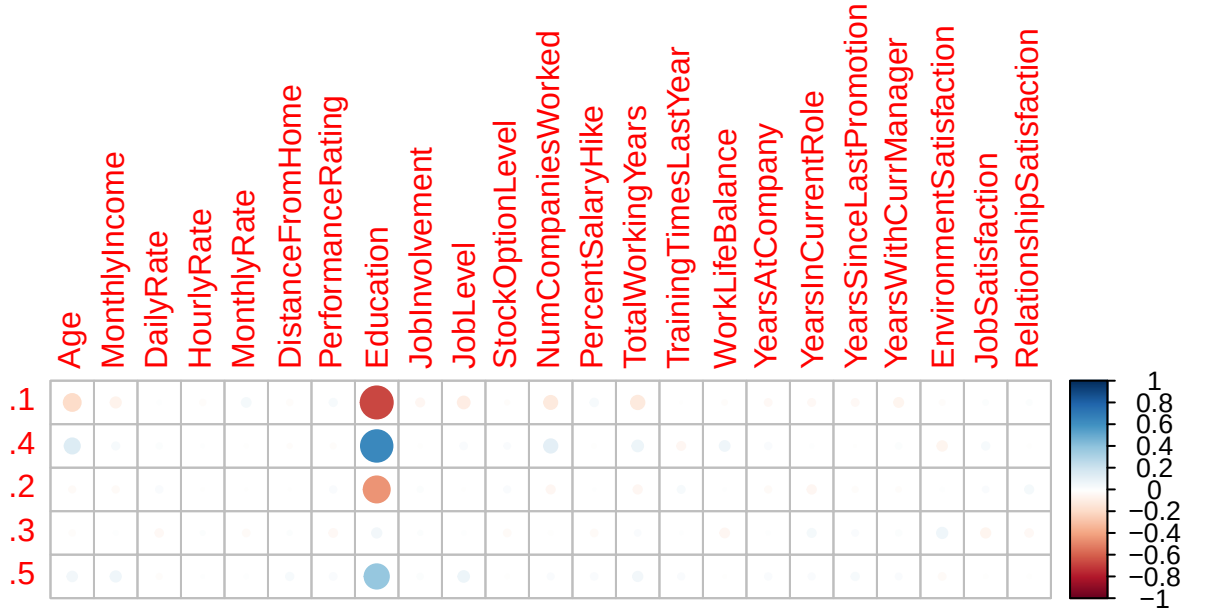


HeatMap

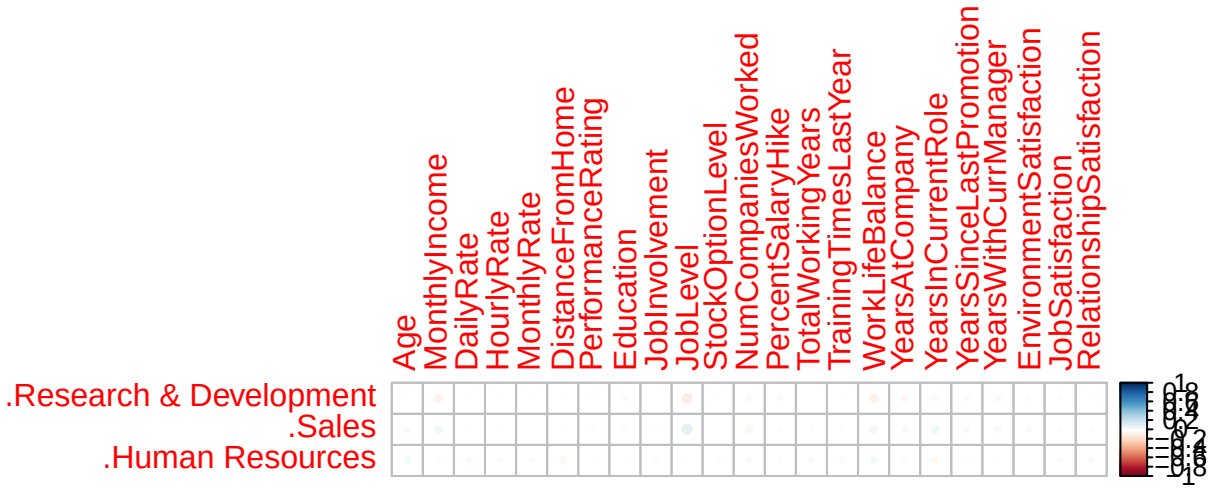
```
color4Var <- prettyGraphs::prettyGraphsColorSelection(ncol(data1))
data2 <- makeNominalData(as.matrix(data[,8]))
corrplot::corrplot(cor(data2,data1))
```




```
data4 <- makeNominalData(as.matrix(data$Education))
corrplot::corrplot(cor(data4,data1))
```

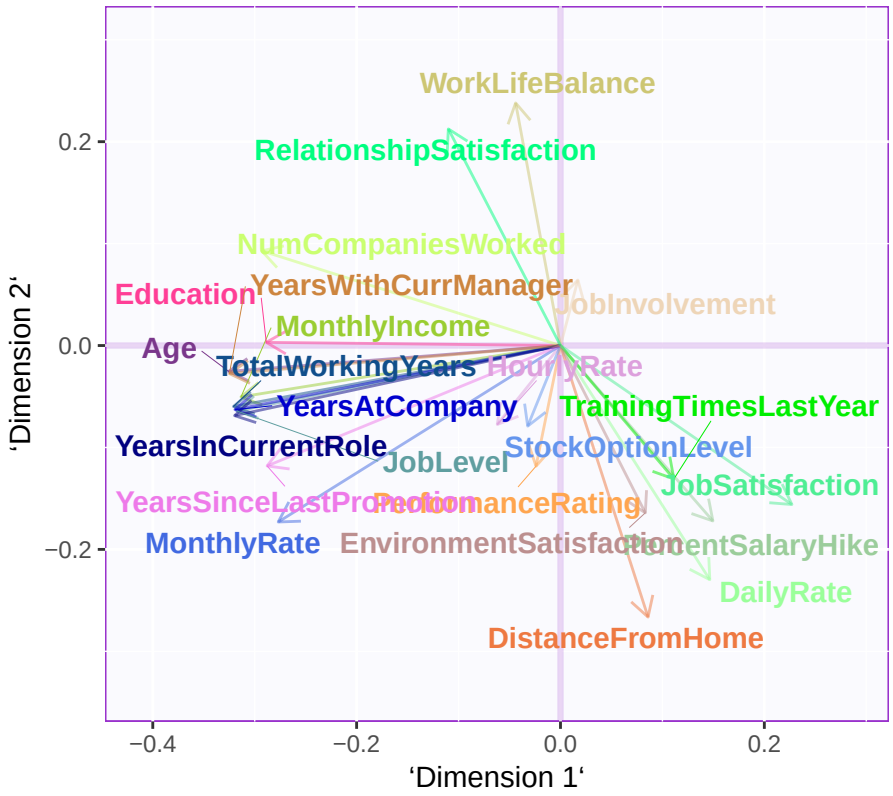



```
data5 <- makeNominalData(as.matrix(data$Department))
corrplot::corrplot(cor(data5,data1))
```



6.1 Factor Map J

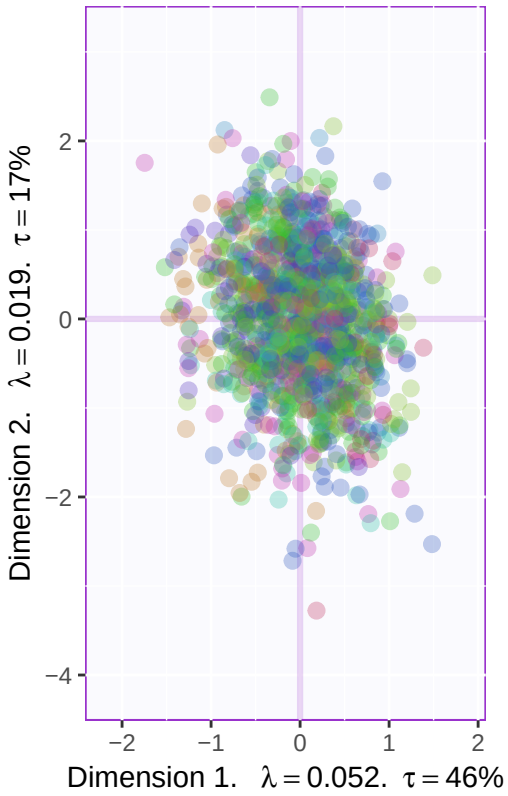
```
col4Var <- prettyGraphsColorSelection(NCOL(data1))
baseMap.j <- PTCA4CATA::createFactorMap(Fj,
                            col.points   = col4Var,
                            alpha.points =  .3,
                            col.labels   = col4Var)
# A graph for the J-set
aggMap.j <- baseMap.j$zeMap_background + # background layer
  baseMap.j$zeMap_dots + baseMap.j$zeMap_text # dots & labels
# We print this Map with the following code
zeLines <- ggplot2::annotate("segment", x = c(0), y = c(0),
                     xend = Fj[,1],
                     yend = Fj[,2],
                     color = col4Var,
                     alpha = .5,
                     arrow = arrow(length = unit(.3, "cm") ) )
# Create the map by adding background, labels, and arrows:
aggMap.j.arrows <- baseMap.j$zeMap_background +
                                      zeLines + baseMap.j$zeMap_text
print(aggMap.j.arrows)
```



6.2 Factor Map I

JobRole

```
baseMap.i <- PTCA4CATA::createFactorMap(Fi,
                                        col.points   = resBADA$Plotting.Data$fii.col,
                                        alpha.points =  .3)
labels4BADA <- createxyLabels.gen(x_axis = 1, y_axis = 2, lambda = resBADA$TExPosition.Data$eigs,
tau = resBADA$TExPosition.Data$t)

# Plain map with color for the I-set
aggMap.i <- baseMap.i$zeMap_background + baseMap.i$zeMap_dots + labels4BADA
#------------------------------------------------------------------------
print(aggMap.i)
```




```
col4data <- resBADA$Plotting.Data$fii.col
col4Means <- unique(col4data)
# create the map for the means
MapGroup     <- PTCA4CATA::createFactorMap(Fk,
                                  axis1 = 1, axis2 = 2,
                                  constraints = baseMap.i$constraints,
                                  title = NULL,
                                  col.points = col4Means,
                                  display.points = TRUE,
                                  pch = 19, cex = 5,
                                  display.labels = TRUE,
                                  col.labels = col4Means,
                                  text.cex = 4,
                                  font.face = "bold",
                                  font.family = "sans",
                                  col.axes = "darkorchid",
                                  alpha.axes = 0.2,
                                  width.axes = 1.1,
                                col.background = adjustcolor("lavender",
                                                             alpha.f = 0.2),
                                force = 1, segment.size = 0)
# The map with observations and group means
aggMap.i.withMeans <- baseMap.i$zeMap_background +
  MapGroup$zeMap_dots + MapGroup$zeMap_text + labels4BADA
#-----------------------------------------------------------------------
print(aggMap.i.withMeans)
```

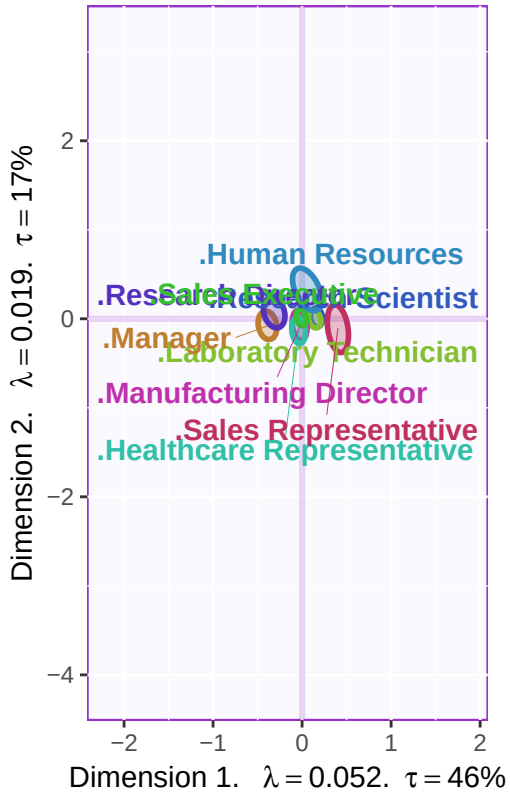
Create 75% Confidence interval polygons

```
GraphTI.Hull.90 <- MakeToleranceIntervals(Fi,
                                  as.factor(datae),
                                  names.of.factors = c("Dim1","Dim2"),
                                  col = unique(col4data),
                                  line.size = .5, line.type = 3,
                                  alpha.ellipse = .2,
                                  alpha.line = .4,
                                  p.level = .75, # 75% TI
                                  type = 'hull' #
                                          # use 'hull' for convex hull
)
#-------------------------------------------------------------------------
# Create the map
aggMap.i.withHull <- baseMap.i$zeMap_background +
  GraphTI.Hull.90 + MapGroup$zeMap_dots +
  MapGroup$zeMap_text +  MapGroup$zeMap_dots + labels4BADA
#-------------------------------------------------------------------------

print(aggMap.i.withHull)
```

Inferences

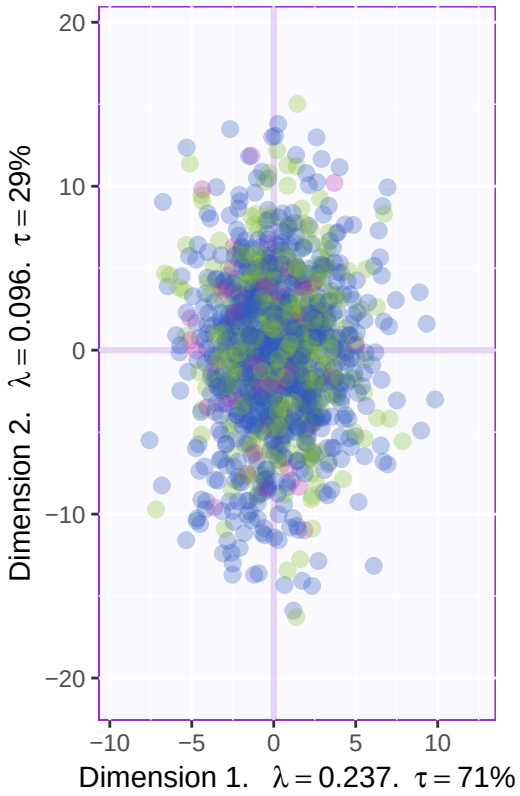
```
resBADA.inf <- tepBADA.inference.battery(DATA = data1,
                    scale = 'SS1', center = TRUE,
                    DESIGN = datae,
                    make_design_nominal = TRUE,
                    group.masses = NULL,
                    weights = NULL,
                    graphs = FALSE,
                    k = 2,
                    test.iters = 100,
                    critical.value = 2)
#----------------------------------------------------------------------
# Confusion matrices
# To be saved as table
fixedCM   <-   resBADA.inf$Inference.Data$loo.data$fixed.confuse
looedCM   <- resBADA.inf$Inference.Data$loo.data$loo.confuse

#----------------------------------------------------------------------
# Create Confidence Interval Plots
BootCube <- resBADA.inf$Inference.Data$boot.data$fi.boot.data$boots
dimnames(BootCube)[[2]] <- c("Dimension 1","Dimension 2")
# use function MakeCIEllipses from package PTCA4CATA
GraphElli <- MakeCIEllipses(BootCube[,1:2,],
                    names.of.factors = c("Dimension 1","Dimension 2"),
                    col = col4Means,
                    p.level = .95
```

```
)
#-------------------------------------------------------------------
# create the I-map with Observations, means and confidence intervals
#
aggMap.i.withCI <-  baseMap.i$zeMap_background +  GraphElli + MapGroup$zeMap_text+ labels4BADA
#-------------------------------------------------------------------

print(aggMap.i.withCI)
```

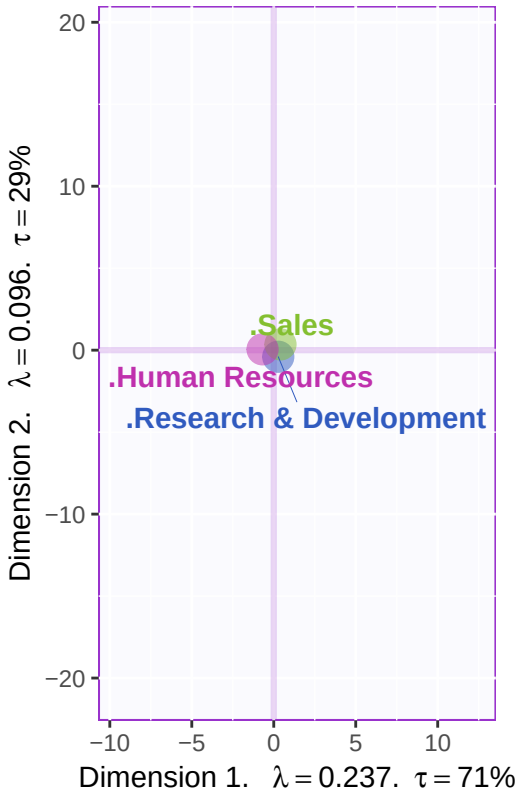


Department

```
baseMap.i1 <- PTCA4CATA::createFactorMap(Fi1,
                                         col.points    = resBADA1$Plotting.Data$fii.col,
                                         alpha.points =  .3)
labels4BADA1 <- createxyLabels.gen(x_axis = 1, y_axis = 2, lambda = resBADA1$TExPosition.Data$eigs,
tau = resBADA1$TExPosition.Data$t)

# Plain map with color for the I-set
aggMap.i1 <- baseMap.i1$zeMap_background + baseMap.i1$zeMap_dots + labels4BADA1
#-------------------------------------------------------------------
print(aggMap.i1)
```

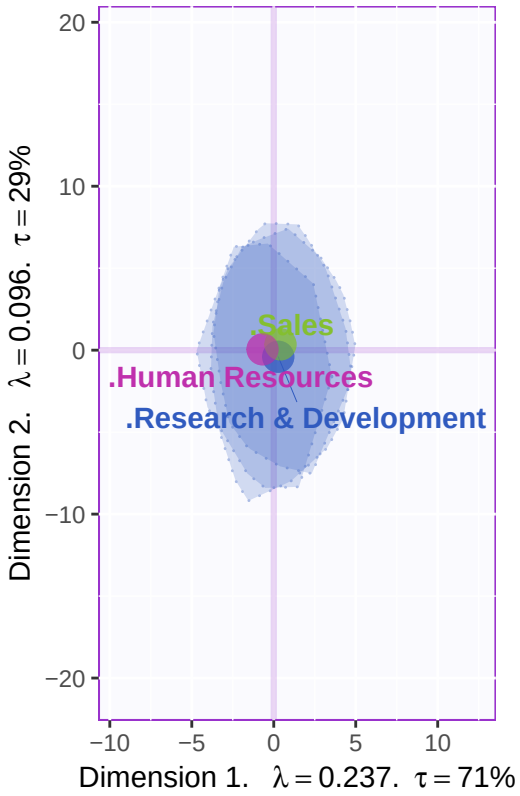



```
col4data1 <- resBADA1$Plotting.Data$fii.col
col4Means1 <- unique(col4data1)
# create the map for the means
MapGroup1    <- PTCA4CATA::createFactorMap(Fk1,
                                axis1 = 1, axis2 = 2,
                                constraints = baseMap.i$constraints,
                                title = NULL,
                                col.points = col4Means1,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means1,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                              col.background = adjustcolor("lavender",
                                                           alpha.f = 0.2),
                              force = 1, segment.size = 0)
# The map with observations and group means
aggMap.i.withMeans1 <- baseMap.i1$zeMap_background +
  MapGroup1$zeMap_dots + MapGroup1$zeMap_text + labels4BADA1
#-----------------------------------------------------------------------
print(aggMap.i.withMeans1)
```



Create 75% Confidence interval polygons

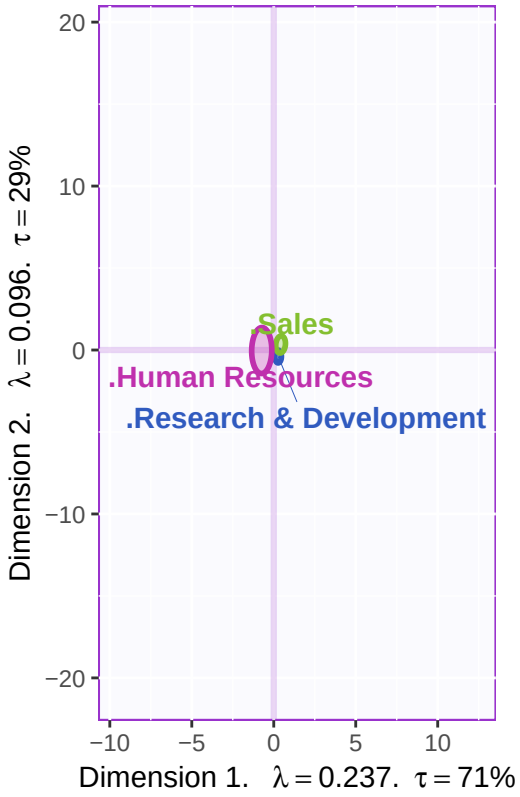
```
GraphTI.Hull.901 <- MakeToleranceIntervals(Fi1,
                                  as.factor(datar),
                                  names.of.factors = c("Dim1","Dim2"),
                                  col = unique(col4data),
                                  line.size = .5, line.type = 3,
                                  alpha.ellipse = .2,
                                  alpha.line = .4,
                                  p.level = .75, # 75% TI
                                  type = 'hull' #
                                          # use 'hull' for convex hull
)
#-------------------------------------------------------------------------
# Create the map
aggMap.i.withHull1 <-  baseMap.i1$zeMap_background  +
  GraphTI.Hull.901 + MapGroup1$zeMap_dots +
  MapGroup1$zeMap_text +  MapGroup1$zeMap_dots + labels4BADA1
#-------------------------------------------------------------------------

print(aggMap.i.withHull1)
```



Inferences

```
resBADA.inf1 <- tepBADA.inference.battery(DATA = data1,
                    scale = 'SS1', center = TRUE,
                    DESIGN = datar,
                    make_design_nominal = TRUE,
                    group.masses = NULL,
                    weights = NULL,
                    graphs = FALSE,
                    k = 2,
                    test.iters = 100,
                    critical.value = 2)
#----------------------------------------------------------------------
# Confusion matrices
# To be saved as table
fixedCM1   <-   resBADA.inf1$Inference.Data$loo.data$fixed.confuse
looedCM1   <- resBADA.inf1$Inference.Data$loo.data$loo.confuse

#----------------------------------------------------------------------
# Create Confidence Interval Plots
BootCube1 <- resBADA.inf1$Inference.Data$boot.data$fi.boot.data$boots
dimnames(BootCube1)[[2]] <- c("Dimension 1","Dimension 2")
# use function MakeCIEllipses from package PTCA4CATA
GraphElli1 <- MakeCIEllipses(BootCube1[,1:2,],
                   names.of.factors = c("Dimension 1","Dimension 2"),
                   col = col4Means1,
                   p.level = .95
```

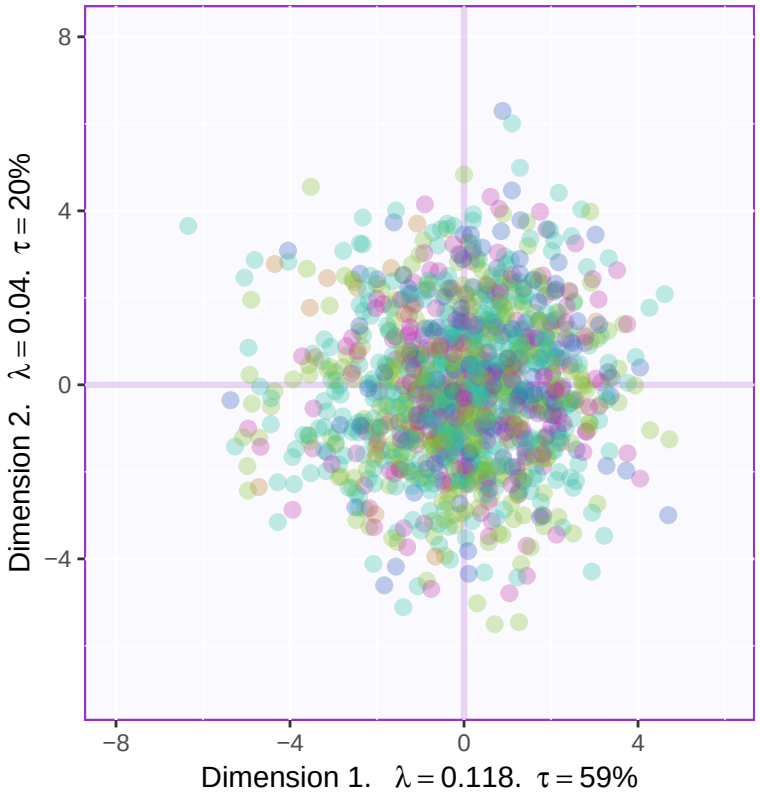
```
)
#----------------------------------------------------------------------
# create the I-map with Observations, means and confidence intervals
#
aggMap.i.withCI1 <-  baseMap.i1$zeMap_background +  GraphElli1 + MapGroup1$zeMap_text+ labels4BADA1
#----------------------------------------------------------------------

print(aggMap.i.withCI1)
```



Education

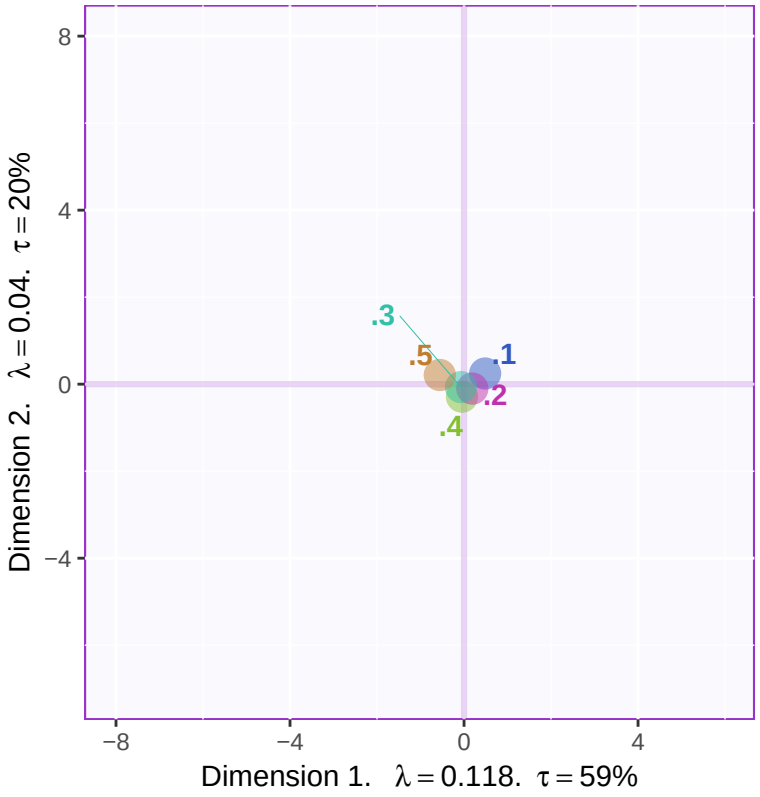
```
resBADA2 <- tepBADA(DATA = data1,
                    scale = 'SS1', center = TRUE,
                    DESIGN = data$Education,
                    make_design_nominal = TRUE,
                    group.masses = NULL,
                    weights = NULL, graphs =  FALSE)
Fk2 <- resBADA2$TExPosition.Data$fi
Fi2 <- resBADA2$TExPosition.Data$fii
Fj2 <- resBADA2$TExPosition.Data$fj

baseMap.i12 <- PTCA4CATA::createFactorMap(Fi2,
                                          col.points    = resBADA2$Plotting.Data$fii.col,
                                          alpha.points =  .3)
labels4BADA12 <- createxyLabels.gen(x_axis = 1, y_axis = 2, lambda = resBADA2$TExPosition.Data$eigs,
tau = resBADA2$TExPosition.Data$t)
```

```
# Plain map with color for the I-set
aggMap.i12 <- baseMap.i12$zeMap_background + baseMap.i12$zeMap_dots + labels4BADA12
#-------------------------------------------------------------------
print(aggMap.i12)
```



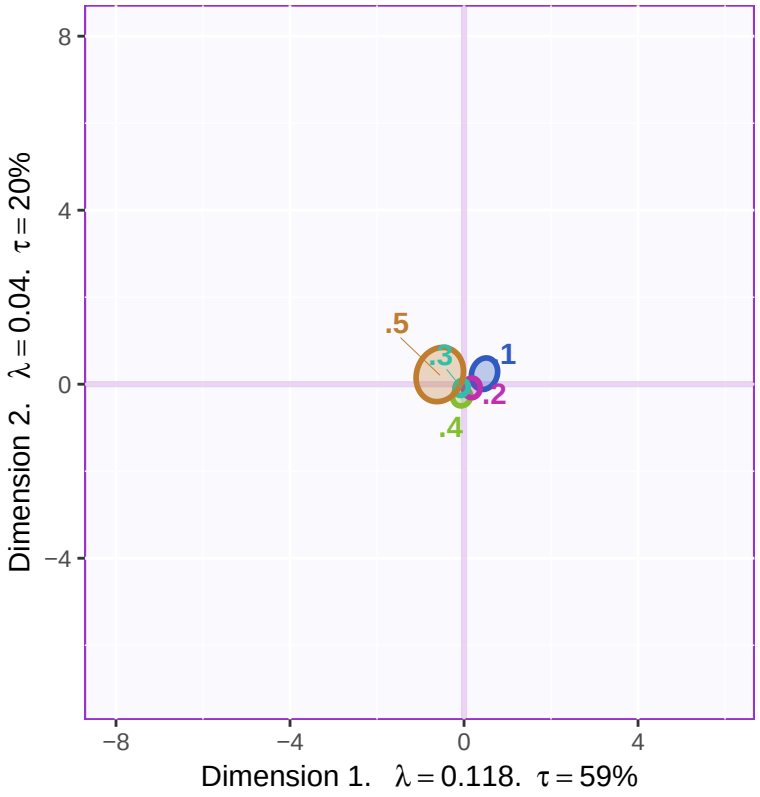

```
col4data2 <- resBADA2$Plotting.Data$fii.col
col4Means2 <- unique(col4data2)
# create the map for the means
MapGroup2   <- PTCA4CATA::createFactorMap(Fk2,
                                axis1 = 1, axis2 = 2,
                                constraints = baseMap.i$constraints,
                                title = NULL,
                                col.points = col4Means2,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means2,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                             col.background = adjustcolor("lavender",
                                                          alpha.f = 0.2),
```

```
                              force = 1, segment.size = 0)
# The map with observations and group means
aggMap.i.withMeans12 <- baseMap.i12$zeMap_background +
  MapGroup2$zeMap_dots + MapGroup2$zeMap_text + labels4BADA12
#-----------------------------------------------------------------------
print(aggMap.i.withMeans12)
```




```
resBADA.inf2 <- tepBADA.inference.battery(DATA = data1,
                   scale = 'SS1', center = TRUE,
                   DESIGN = data$Education,
                   make_design_nominal = TRUE,
                   group.masses = NULL,
                   weights = NULL,
                   graphs = FALSE,
                   k = 2,
                   test.iters = 100,
                   critical.value = 2)
#-----------------------------------------------------------------------
# Confusion matrices
# To be saved as table
fixedCM2   <-   resBADA.inf2$Inference.Data$loo.data$fixed.confuse
looedCM2   <- resBADA.inf2$Inference.Data$loo.data$loo.confuse

#-----------------------------------------------------------------------
# Create Confidence Interval Plots
BootCube2 <- resBADA.inf2$Inference.Data$boot.data$fi.boot.data$boots
```

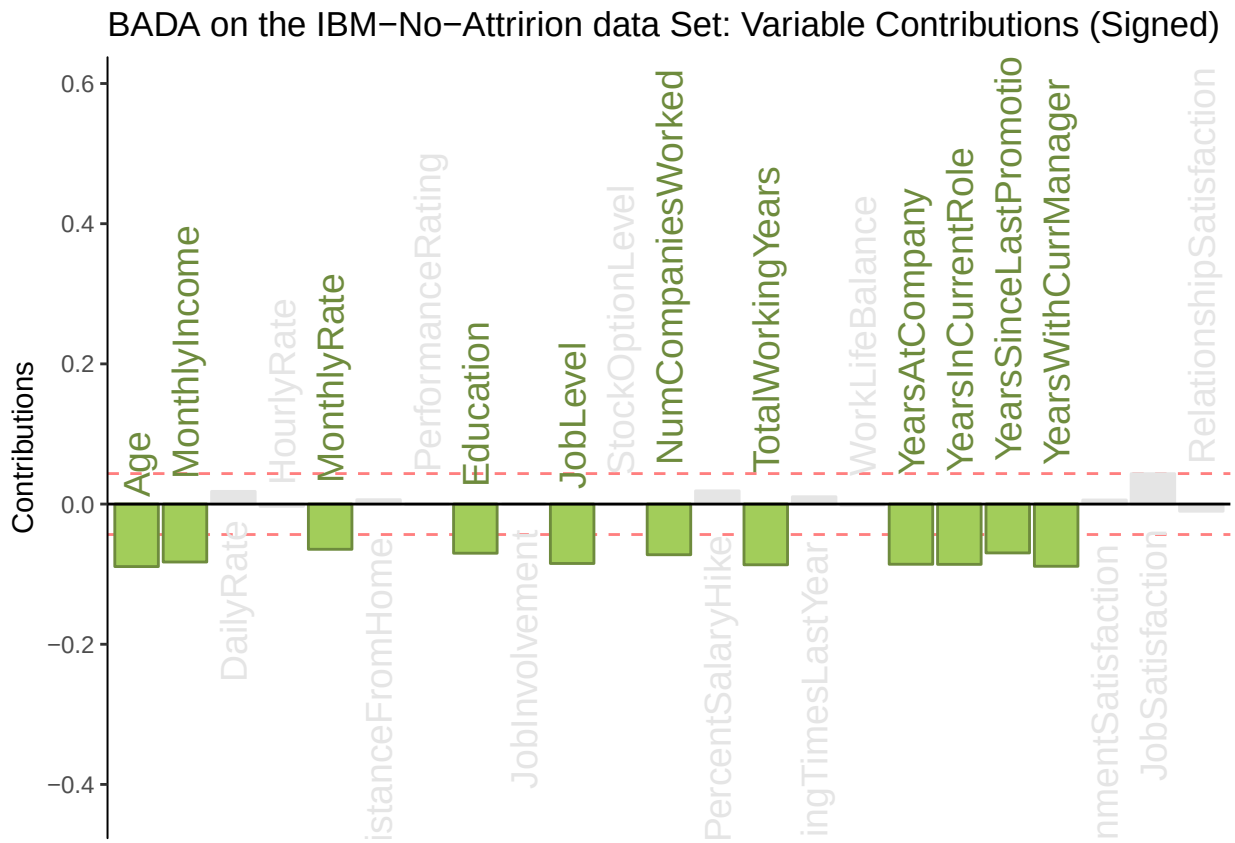
```
dimnames(BootCube2)[[2]] <- c("Dimension 1","Dimension 2")
# use function MakeCIEllipses from package PTCA4CATA
GraphElli2 <- MakeCIEllipses(BootCube2[,1:2,],
                  names.of.factors = c("Dimension 1","Dimension 2"),
                  col = col4Means2,
                  p.level = .95
)
#-----------------------------------------------------------------------
# create the I-map with Observations, means and confidence intervals
#
aggMap.i.withCI12 <-  baseMap.i12$zeMap_background +  GraphElli2 + MapGroup2$zeMap_text+ labels4BADA12
#-----------------------------------------------------------------------

print(aggMap.i.withCI12)
```



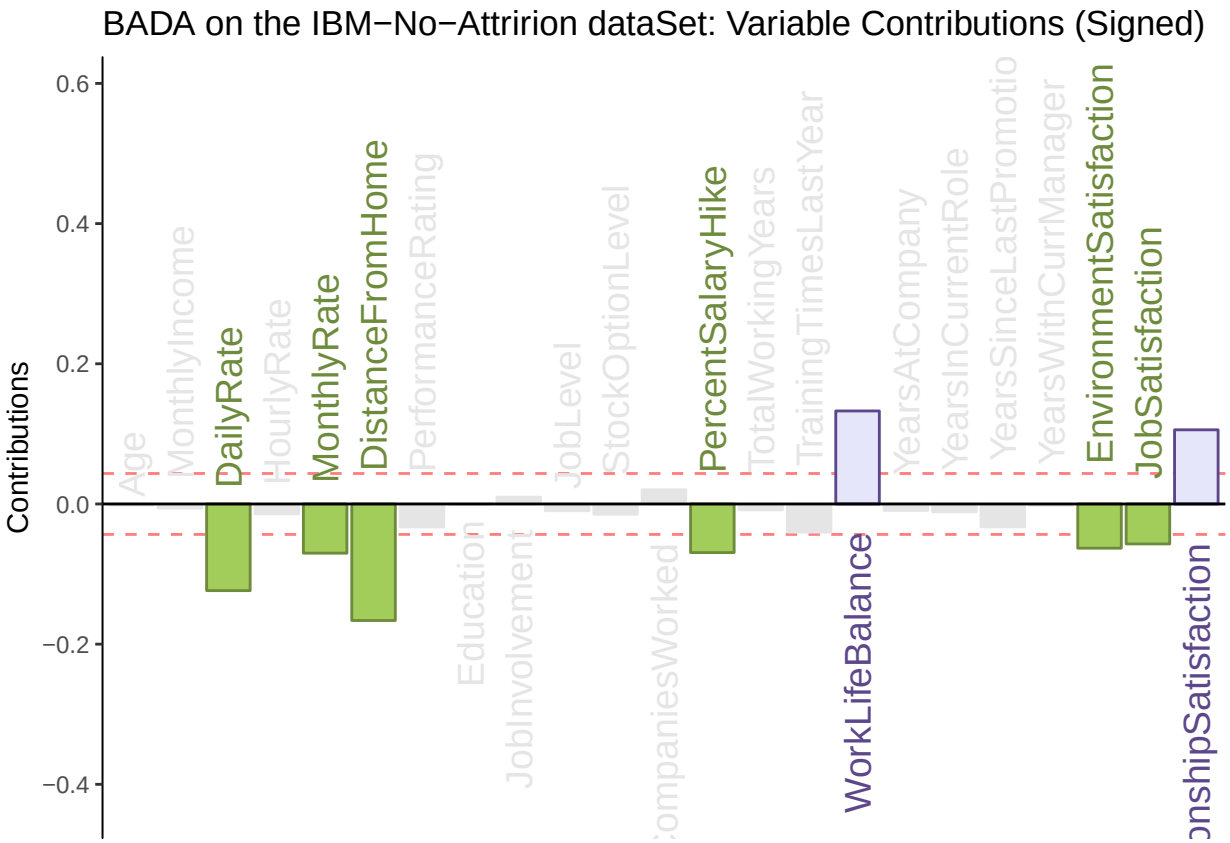
6.3 Contribution

```
signed.ctrJ1 <- resBADA$TExPosition.Data$cj * sign(resBADA$TExPosition.Data$fj)
b003.ctrJ.s.11 <- PrettyBarPlot2(signed.ctrJ1[,1],
                                 threshold = 1 / NROW(signed.ctrJ1),
                                 font.size = 5,
                                 # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                 main = 'BADA on the IBM-No-Attririon data Set: Variable Contributions (S
                                 ylab = 'Contributions',
```

```
                                ylim = c(1.2*min(signed.ctrJ1), 1.2*max(signed.ctrJ1))
)
print(b003.ctrJ.s.11)
```

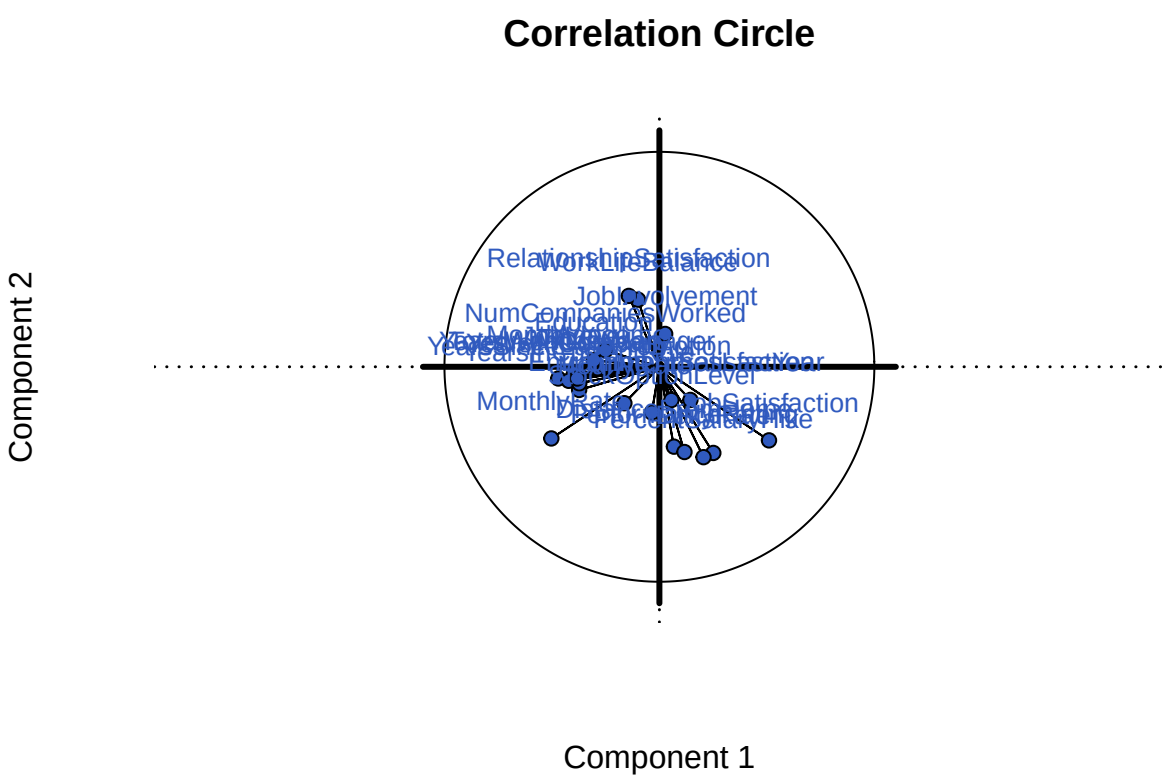



```
b004.ctrJ.s.21 <- PrettyBarPlot2(signed.ctrJ1[,2],
                                 threshold = 1 / NROW(signed.ctrJ1),
                                 font.size = 5,
                                 # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                 main = 'BADA on the IBM-No-Attririon dataSet: Variable Contributions (S
                                 ylab = 'Contributions',
                                 ylim = c(1.2*min(signed.ctrJ1), 1.2*max(signed.ctrJ1))
)
print(b004.ctrJ.s.21)
```



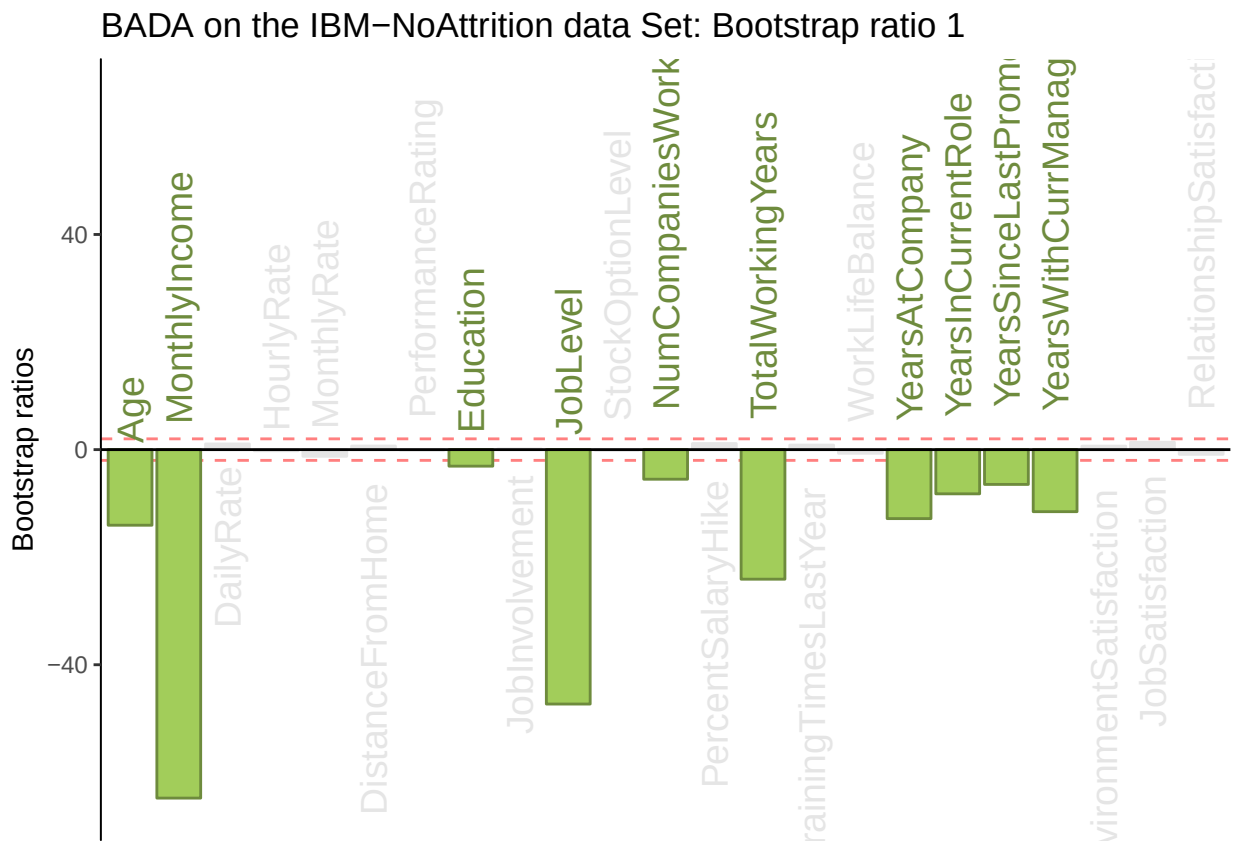
Correlation Circle

```
correlationCircle <- correlationPlotter(data_matrix = data1 , factor_scores = resBADA$TExPosition.Data$
```



6.4 Bootstrap Ratios

```
BR1 <- resBADA.inf$Inference.Data$boot.data$fj.boot.data$tests$boot.ratios
laDim = 1
ba001.BR11 <- PrettyBarPlot2(BR1[,laDim],
                             threshold = 2,
                             font.size = 5,
                             #color4bar = gplots::col2hex(col4J.ibm),
                             main = paste0( 'BADA on the IBM-NoAttrition data Set: Bootstrap ratio ',laD
                             ylab = 'Bootstrap ratios'
                             #ylim = c(1.2*min(BR[,laDim]), 1.2*max(BR[,laDim]))
)
print(ba001.BR11)
```




```
#
laDim = 2
ba002.BR21 <- PrettyBarPlot2(BR1[,laDim],
                             threshold = 2,
                             font.size = 5,
                             #color4bar = gplots::col2hex(col4J.ibm),
                             main = paste0(
                               'BADA on the IBM-NoAttrition data Set: Bootstrap ratio ',laDim),
                             ylab = 'Bootstrap ratios'
)
print(ba002.BR21)
```

Random (LOO) confusion matrix

```
#-----------------------------------------------------------------------
# Confusion matrices
# To be saved as table
print(as.data.frame(fixedCM))
```

```
##                            .Research Scientist .Laboratory Technician
## .Research Scientist                          5                      17
## .Laboratory Technician                       0                       1
## .Manufacturing Director                      4                       2
## .Healthcare Representative                  39                      29
## .Manager                                    19                      18
## .Research Director                          16                      11
## .Sales Executive                             5                       4
## .Sales Representative                       74                      59
## .Human Resources                            83                      56
##                            .Manufacturing Director
## .Research Scientist                              7
## .Laboratory Technician                           2
## .Manufacturing Director                          0
## .Healthcare Representative                      32
## .Manager                                        14
## .Research Director                              18
## .Sales Executive                                 3
## .Sales Representative                           25
## .Human Resources                                34
```

```
##                            .Healthcare Representative .Manager
## .Research Scientist                                 1        1
## .Laboratory Technician                              1        1
## .Manufacturing Director                             1        2
## .Healthcare Representative                         26       18
## .Manager                                           21       26
## .Research Director                                  7       20
## .Sales Executive                                    2        1
## .Sales Representative                              22        9
## .Human Resources                                   41       19
##                            .Research Director .Sales Executive
## .Research Scientist                         1                6
## .Laboratory Technician                      3                2
## .Manufacturing Director                     1                2
## .Healthcare Representative                  7               42
## .Manager                                   22               38
## .Research Director                         16               33
## .Sales Executive                            2                2
## .Sales Representative                      10               60
## .Human Resources                           16               84
##                            .Sales Representative .Human Resources
## .Research Scientist                            1                0
## .Laboratory Technician                         0                1
## .Manufacturing Director                        2                0
## .Healthcare Representative                     3                2
## .Manager                                       0                2
## .Research Director                             0                5
## .Sales Executive                               0                0
## .Sales Representative                         25               11
## .Human Resources                              19               19
```

```
print(as.data.frame(looedCM))
```

```
##                                      .Research Scientist.actual
## .Research Scientist.predicted                                 5
## .Laboratory Technician.predicted                              0
## .Manufacturing Director.predicted                             4
## .Healthcare Representative.predicted                         39
## .Manager.predicted                                           19
## .Research Director.predicted                                 16
## .Sales Executive.predicted                                    5
## .Sales Representative.predicted                              73
## .Human Resources.predicted                                   84
##                                      .Laboratory Technician.actual
## .Research Scientist.predicted                                   16
## .Laboratory Technician.predicted                                 2
## .Manufacturing Director.predicted                                2
## .Healthcare Representative.predicted                            27
## .Manager.predicted                                              20
## .Research Director.predicted                                    14
## .Sales Executive.predicted                                       3
## .Sales Representative.predicted                                 58
## .Human Resources.predicted                                      55
##                                      .Manufacturing Director.actual
## .Research Scientist.predicted                                     7
```

```
## .Laboratory Technician.predicted                                3
## .Manufacturing Director.predicted                                0
## .Healthcare Representative.predicted                            30
## .Manager.predicted                                              14
## .Research Director.predicted                                    15
## .Sales Executive.predicted                                       2
## .Sales Representative.predicted                                 25
## .Human Resources.predicted                                      39
##                                    .Healthcare Representative.actual
## .Research Scientist.predicted                                      1
## .Laboratory Technician.predicted                                   2
## .Manufacturing Director.predicted                                  0
## .Healthcare Representative.predicted                              24
## .Manager.predicted                                                21
## .Research Director.predicted                                       6
## .Sales Executive.predicted                                         3
## .Sales Representative.predicted                                   23
## .Human Resources.predicted                                        42
##                                    .Manager.actual
## .Research Scientist.predicted                    1
## .Laboratory Technician.predicted                 0
## .Manufacturing Director.predicted                0
## .Healthcare Representative.predicted            19
## .Manager.predicted                              21
## .Research Director.predicted                    20
## .Sales Executive.predicted                       2
## .Sales Representative.predicted                 12
## .Human Resources.predicted                      22
##                                    .Research Director.actual
## .Research Scientist.predicted                              3
## .Laboratory Technician.predicted                           1
## .Manufacturing Director.predicted                          1
## .Healthcare Representative.predicted                       8
## .Manager.predicted                                        22
## .Research Director.predicted                              13
## .Sales Executive.predicted                                 1
## .Sales Representative.predicted                           13
## .Human Resources.predicted                                16
##                                    .Sales Executive.actual
## .Research Scientist.predicted                             5
## .Laboratory Technician.predicted                          2
## .Manufacturing Director.predicted                         2
## .Healthcare Representative.predicted                     42
## .Manager.predicted                                       38
## .Research Director.predicted                             33
## .Sales Executive.predicted                                4
## .Sales Representative.predicted                          61
## .Human Resources.predicted                               82
##                                    .Sales Representative.actual
## .Research Scientist.predicted                                 1
## .Laboratory Technician.predicted                              0
## .Manufacturing Director.predicted                             1
## .Healthcare Representative.predicted                          7
## .Manager.predicted                                            0
```

```
## .Research Director.predicted                                    1
## .Sales Executive.predicted                                      1
## .Sales Representative.predicted                                18
## .Human Resources.predicted                                     21
##                                      .Human Resources.actual
## .Research Scientist.predicted                                  1
## .Laboratory Technician.predicted                               0
## .Manufacturing Director.predicted                              0
## .Healthcare Representative.predicted                           7
## .Manager.predicted                                             4
## .Research Director.predicted                                   5
## .Sales Executive.predicted                                     0
## .Sales Representative.predicted                               13
## .Human Resources.predicted                                    10
```

```
#----------------------------------------------------------------------
# Confusion matrices
# To be saved as table
fixedCM1   <-   resBADA.inf1$Inference.Data$loo.data$fixed.confuse
looedCM1   <- resBADA.inf1$Inference.Data$loo.data$loo.confuse
print(as.data.frame(fixedCM1))
```

```
##                         .Research & Development .Sales .Human Resources
## .Research & Development                     323    125               19
## .Sales                                      282    148               14
## .Human Resources                            223     81               18
```

```
print(as.data.frame(looedCM1))
```

```
##                                  .Research & Development.actual
## .Research & Development.predicted                            314
## .Sales.predicted                                             283
## .Human Resources.predicted                                   231
##                                  .Sales.actual .Human Resources.actual
## .Research & Development.predicted           127                      16
## .Sales.predicted                            147                      14
## .Human Resources.predicted                   80                      21
```

```
#----------------------------------------------------------------------
# Confusion matrices
# To be saved as table
print(as.data.frame(fixedCM2))
```

```
##    .1  .4 .2  .3 .5
## .1 73 107 86 162 13
## .4 27 101 65 138 11
## .2  5  17 19  34  2
## .3  2   5  2   3  1
## .5 32 110 66 136 16
```

```
print(as.data.frame(looedCM2))
```

```
##              .1.actual .4.actual .2.actual .3.actual .5.actual
## .1.predicted        67       110        86       161        12
## .4.predicted        32        96        64       134        18
## .2.predicted         9        17        17        35         2
## .3.predicted         0         5         4         4         0
```

```
## .5.predicted        31       112        67       139        11
```

6.5 Summary

Since, The goal of BADA is to combine the measurements to create new variables (called components or discriminant variables) that best separate the categories and further are also used to assign the original observations or new observations to the a-priori defined categories.You can see from the above confusion matrix that this method works poorly on this dataset and in comparison to PCA the group means are more closer to each other telling that this method is not the optimal method for the discrimination of the groups. Althought certain inferences could be observed if have to:

- Generally, Managers and Directors are the ones with PhD & Master's with higher job level, Sales representative and Lab Technician have no college education while Healthcare Representatives & Manufacturing Directors have Bachelor's degree
- People who are Managers and Research Directors have the highest pay and are more seasoned while people who are Sales representative and Lab Technician have low pay and less experience
- It is also observed that the Mangers and Research Directors travel the most, could be their meetings and conferences while the HR, Sales representative,Lab Technician, Research Scientists travel occasionally

```
options(knitr.duplicate.label = 'allow')
```

7 DiCA

The main idea behind DCA is to represent each group by the sum of its observations and to perform a simple CA on the groups by variables matrix. The original observations are then projected as supplementary elements and each observation is assigned to the closest group. The comparison between the a priori and the a posteriori classifications can be used to assess the quality of the discrimination. A similar procedure can be used to assign new observations to categories. The stability of the analysis can be evaluated using cross-validation techniques such as jackknifing or bootstrapping.

```
options(knitr.duplicate.label = 'allow')
my_data <- read.csv("IBM-HR-Emplyee-NoAttrition.csv")
cols <- colnames(my_data)
rownames(my_data) <- my_data$Subj
data1 <- my_data[,11:32]
data1$Age <- my_data$Age
```

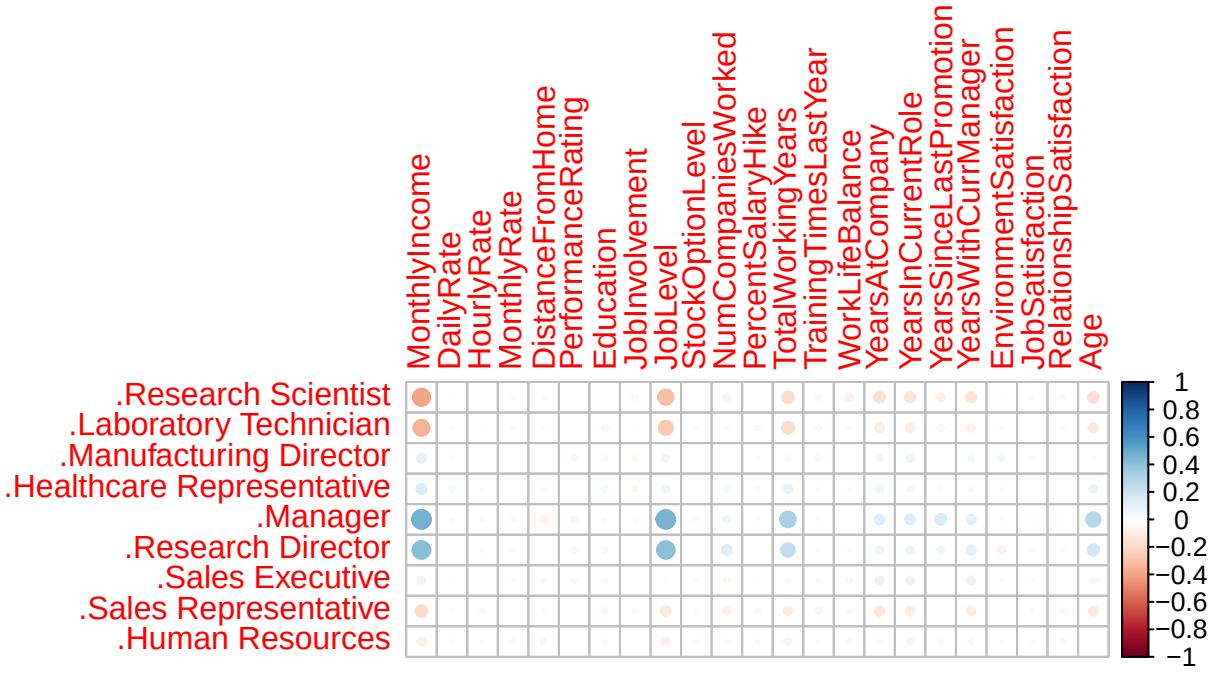
Distribution of the quantitative data

Binning the quantitative data in the similar fashion as MCA

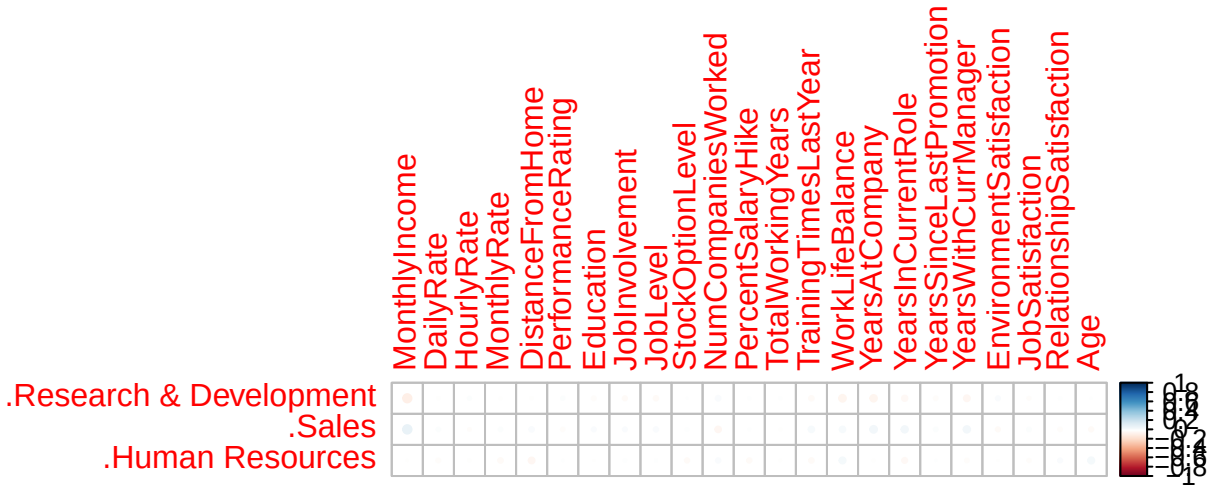
The goal of binning the quantitative data is to convert them into nominal scale so that presence of non-linear realtionship in data can be detected. Also, binning is usually done by diving the variable into equal number of bins of tables or the correlation between the divided bins of tables should be greater than 0.80

7.1 Heatmap

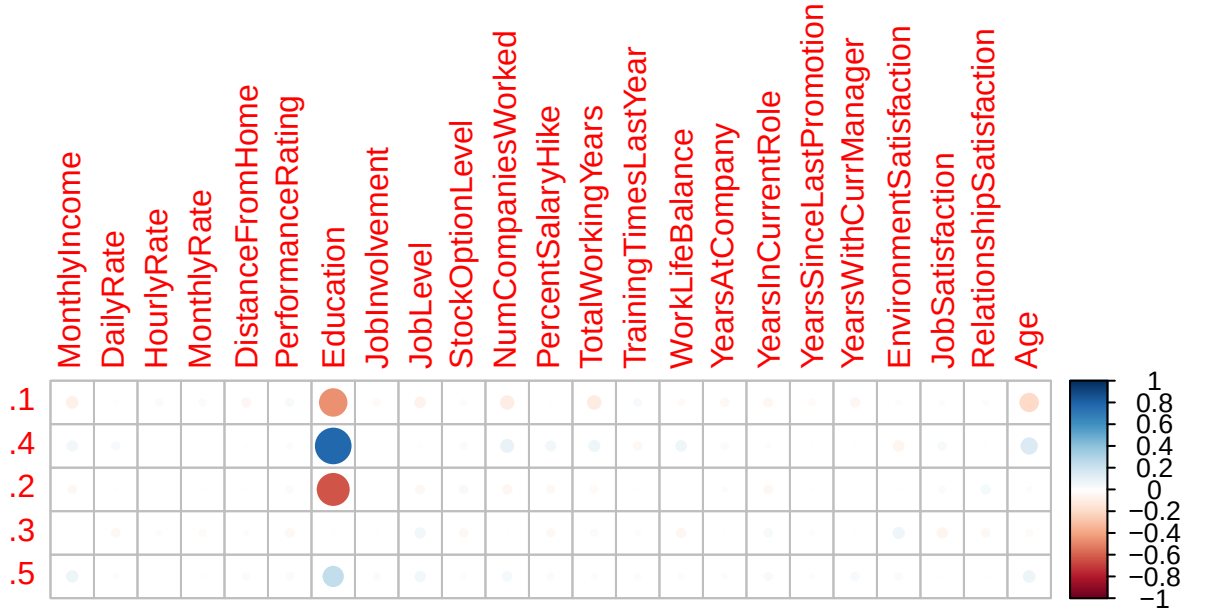
```
color4Var <- prettyGraphs::prettyGraphsColorSelection(ncol(data1))
data2 <- makeNominalData(as.matrix(my_data[,8]))
data3<- data.matrix(data1)
corrplot::corrplot(cor(data2,data3))
```




```
data4 <- makeNominalData(as.matrix(my_data[,5]))
corrplot::corrplot(cor(data4,data3))
```




```
data5 <- makeNominalData(as.matrix(my_data$Education))
corrplot::corrplot(cor(data5,data3))
```




```
data6 <- makeNominalData(as.matrix(my_data$BusinessTravel))
corrplot::corrplot(cor(data6,data3))
```


```
datae <- my_data$JobRole
datar <- my_data$Department
resDICA <- tepDICA(DATA = data1,
                   DESIGN = datae,
                   make_design_nominal = TRUE,
                   make_data_nominal = TRUE,
                   group.masses = NULL,
                   weights = NULL, graphs =  FALSE)

resDICA1 <- tepDICA(DATA = data1,
                   DESIGN = datar,
                   make_design_nominal = TRUE,
                   make_data_nominal = TRUE,
                   group.masses = NULL,
                   weights = NULL, graphs =  FALSE)
```

```
dataedu <- my_data$Education
datatravel <- my_data$BusinessTravel
resDICA2 <- tepDICA(DATA = data1,
                   DESIGN = dataedu,
                   make_design_nominal = TRUE,
                   make_data_nominal = TRUE,
                   group.masses = NULL,
                   weights = NULL, graphs =  FALSE)

resDICA3 <- tepDICA(DATA = data1,
```

```
                    DESIGN = datatravel,
                    make_design_nominal = TRUE,
                    make_data_nominal = TRUE,
                    group.masses = NULL,
                    weights = NULL, graphs =  FALSE)
```

```
datag <- my_data$Gender
resDICA4 <- tepDICA(DATA = data1,
                    DESIGN = datag,
                    make_design_nominal = TRUE,
                    make_data_nominal = TRUE,
                    group.masses = NULL,
                    weights = NULL, graphs =  FALSE)
```

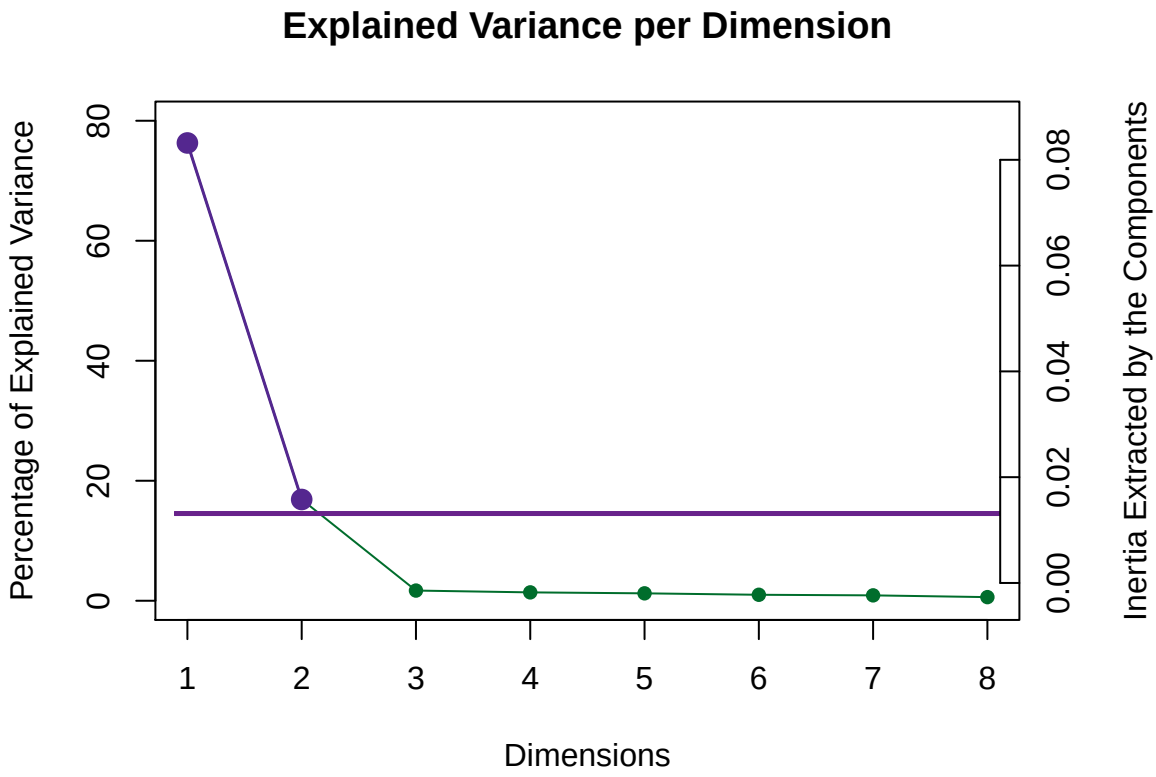
```
## Warning in genPDQ(datain = mRP$deviations, M = mRP$masses, W =
## mRP$weights, : Solution has only 1 singular value/vector. Zeros are
## appended for plotting purposes.
```

ScreePlot for DICA

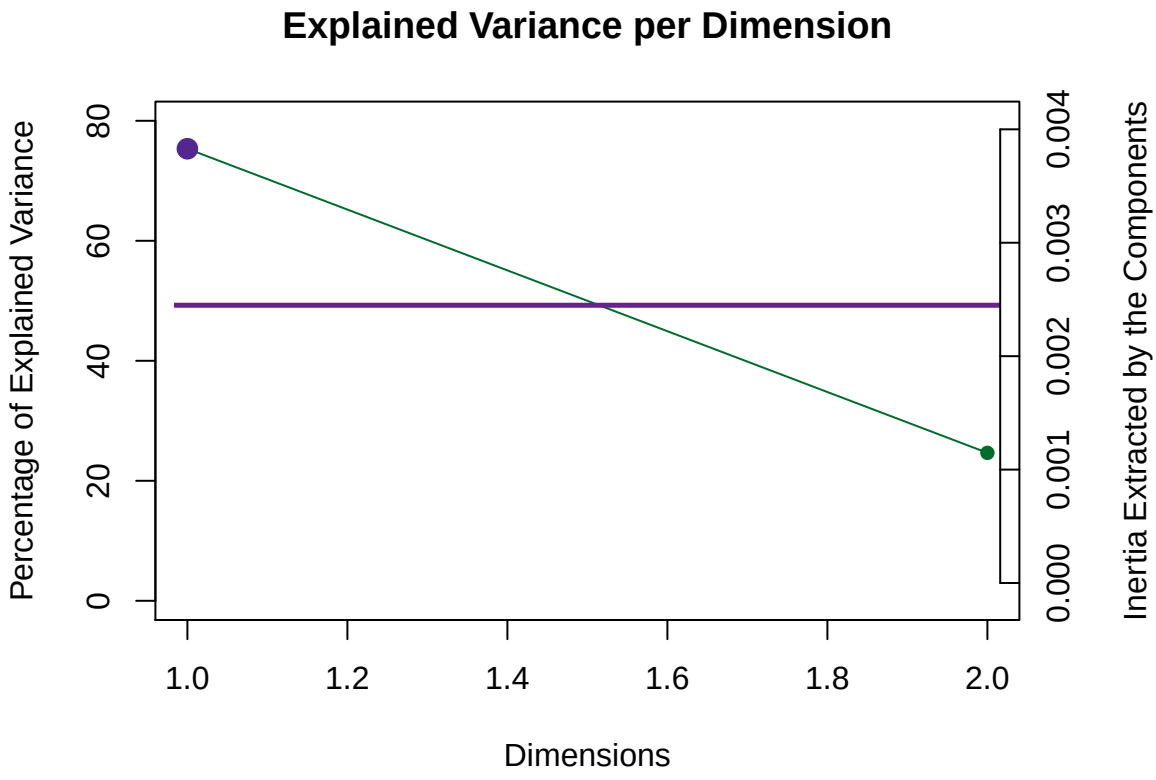
```
resDICA.inf1 <- tepDICA.inference.battery(DATA = data1,
                    DESIGN = datae,
                    make_design_nominal = TRUE,
                    make_data_nominal = TRUE,
                    group.masses = NULL,
                    weights = NULL,
                    graphs = FALSE,
                    k = 2,
                    test.iters = 100,
                    critical.value = 2)
resDICA.inf2 <- tepDICA.inference.battery(DATA = data1,
                    DESIGN = dataedu,
                    make_design_nominal = TRUE,
                    make_data_nominal = TRUE,
                    group.masses = NULL,
                    weights = NULL,
                    graphs = FALSE,
                    k = 2,
                    test.iters = 100,
                    critical.value = 2)
resDICA.inf3 <- tepDICA.inference.battery(DATA = data1,
                    DESIGN = datatravel,
                    make_design_nominal = TRUE,
                    make_data_nominal = TRUE,
                    group.masses = NULL,
                    weights = NULL,
                    graphs = FALSE,
                    k = 2,
                    test.iters = 100,
                    critical.value = 2)
resDICA.in4 <- tepDICA.inference.battery(DATA = data1,
                    DESIGN = datar,
                    make_design_nominal = TRUE,
                    make_data_nominal = TRUE,
                    group.masses = NULL,
```

```
                    weights = NULL,
                    graphs = FALSE,
                    k = 2,
                    test.iters = 100,
                    critical.value = 2)
```

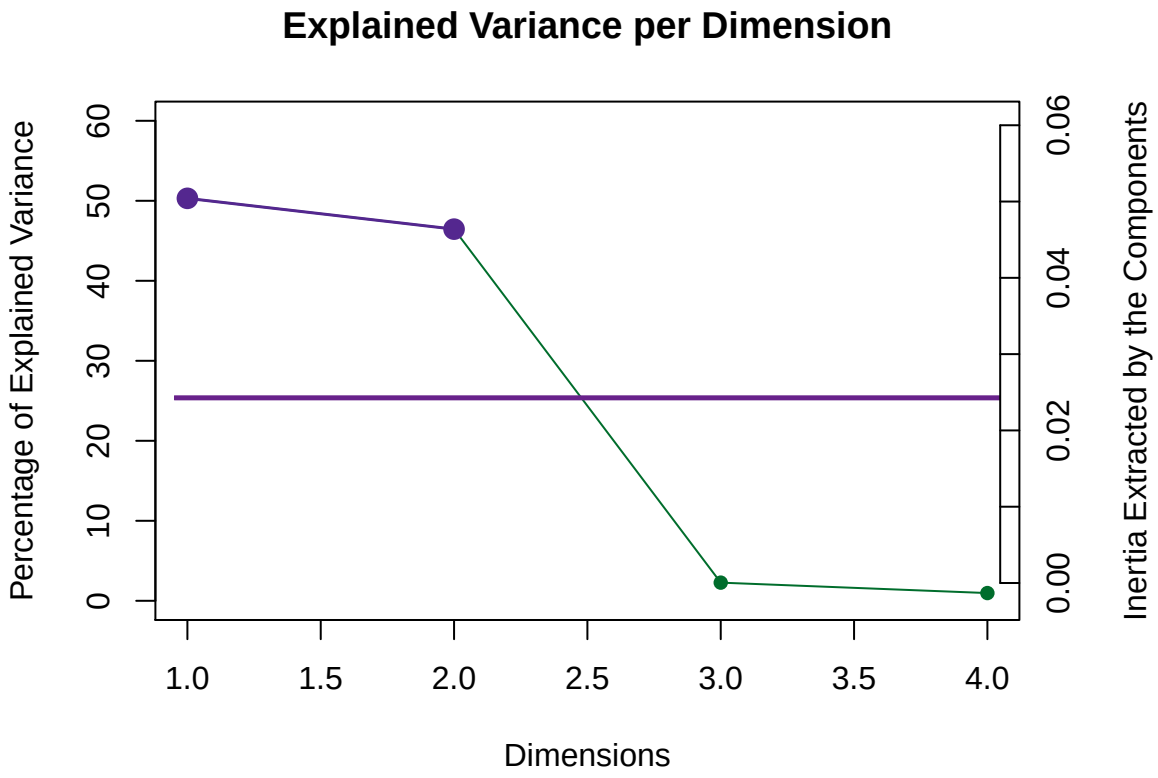
```
PlotScree(ev = resDICA$TExPosition.Data$eigs,
          p.ev = resDICA.inf1$Inference.Data$components$p.vals, max.ev = NULL, alpha = 0.05,
          col.ns = "#006D2C", col.sig = "#54278F",
          title = "Explained Variance per Dimension",plotKaiser = TRUE)
```




```
PlotScree(ev = resDICA1$TExPosition.Data$eigs,
          p.ev = resDICA.in4$Inference.Data$components$p.vals, max.ev = NULL, alpha = 0.05,
          col.ns = "#006D2C", col.sig = "#54278F",
          title = "Explained Variance per Dimension",plotKaiser = TRUE)
```

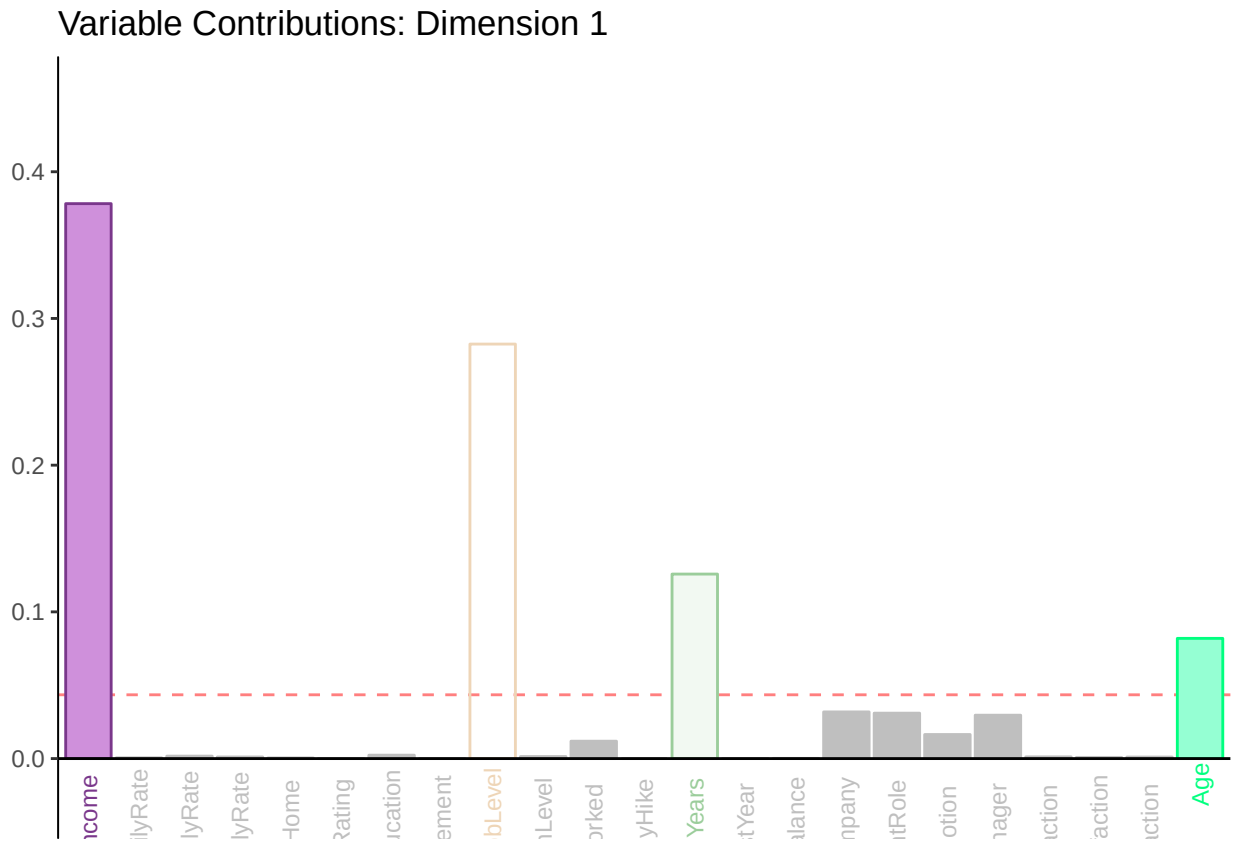



```
PlotScree(ev = resDICA2$TExPosition.Data$eigs,
          p.ev = resDICA.inf2$Inference.Data$components$p.vals, max.ev = NULL, alpha = 0.05,
          col.ns = "#006D2C", col.sig = "#54278F",
          title = "Explained Variance per Dimension",plotKaiser = TRUE)
```

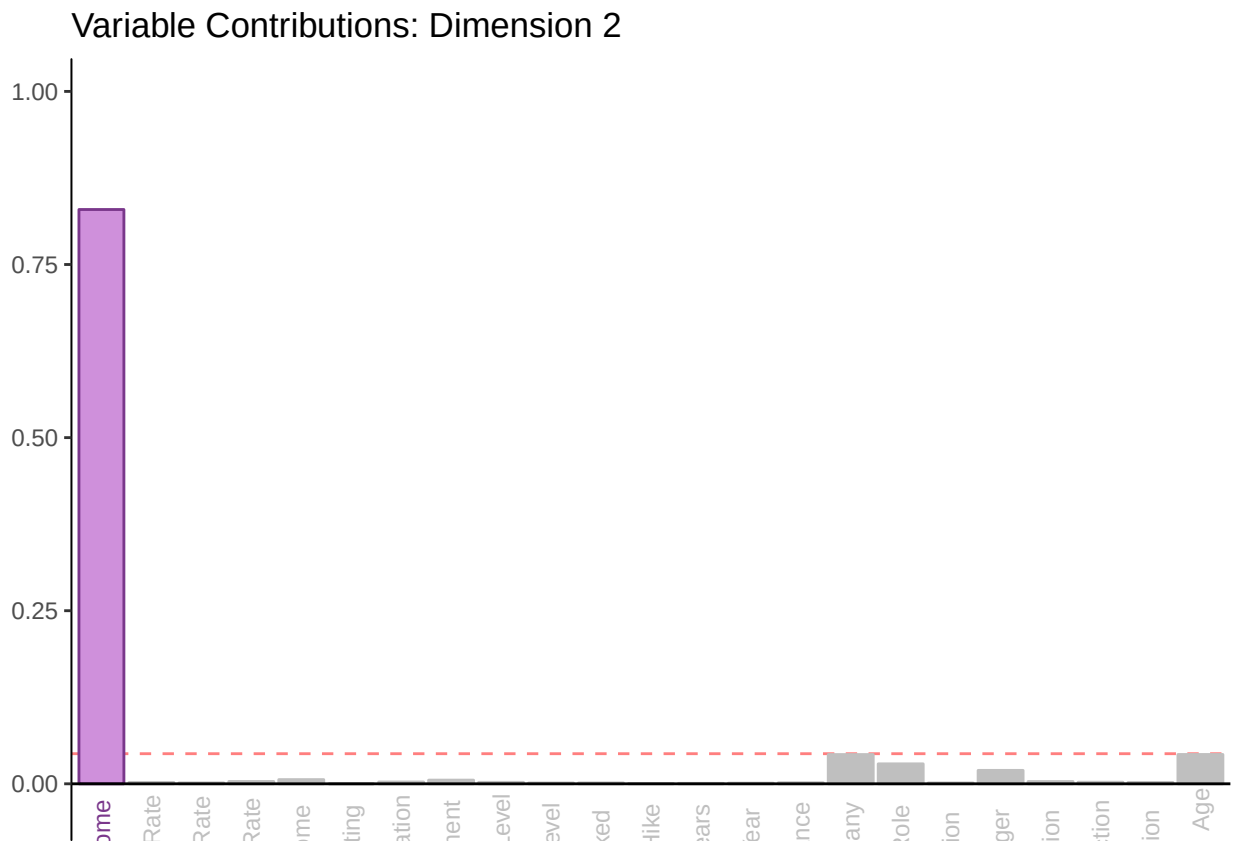


7.2 Variable contributions

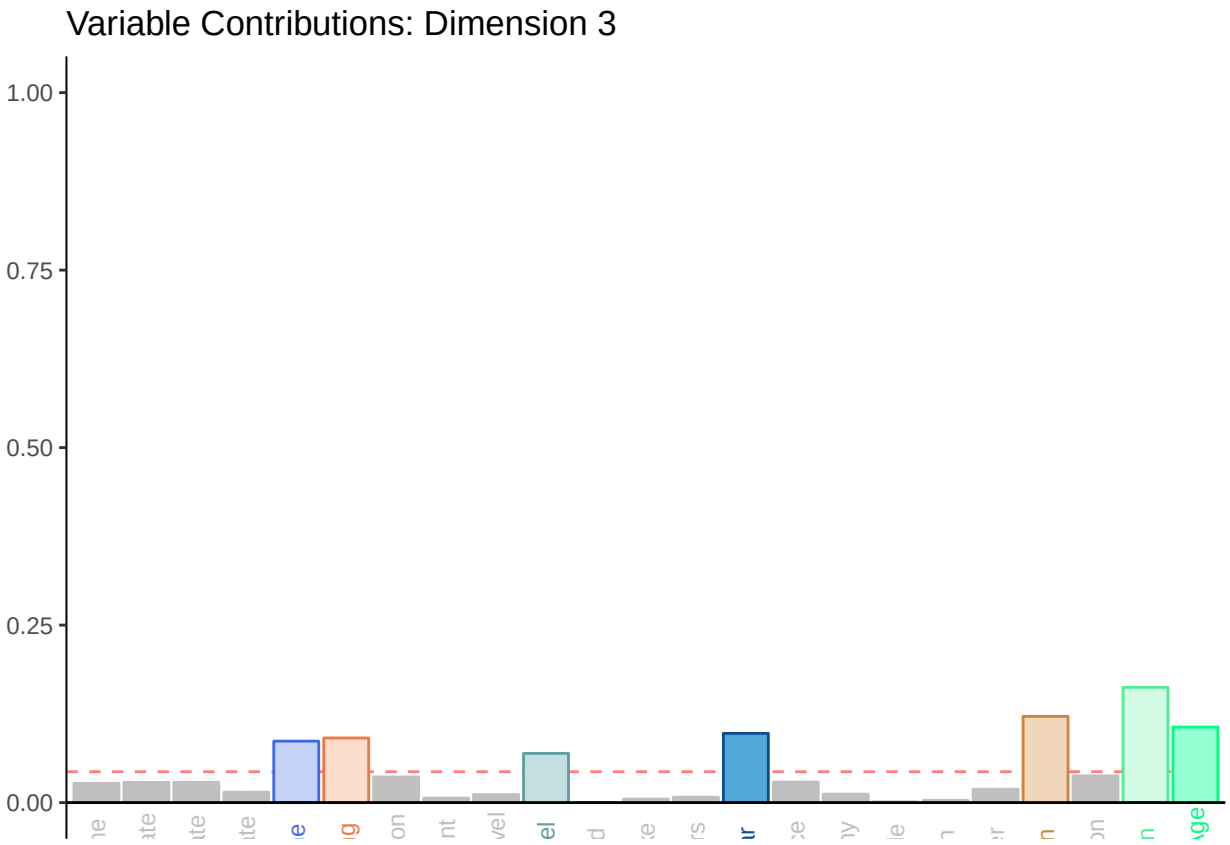
```
color <- prettyGraphs::prettyGraphsColorSelection(ncol(data1))
cJ <- resDICA$TExPosition.Data$cj
varCtr <- data4PCCAR::ctr4Variables(cJ)
rownames(color) <- rownames(varCtr)
varCtr1 <- varCtr[,1]
names(varCtr1) <- rownames(varCtr)
a0005.Var.ctr1 <- PrettyBarPlot2(varCtr1, main = 'Variable Contributions: Dimension 1', ylim = c(-.03,1
print(a0005.Var.ctr1)
```




```
varCtr2 <- varCtr[,2]
names(varCtr2) <- rownames(varCtr)
a0006.Var.ctr2 <- PrettyBarPlot2(varCtr2,
main = 'Variable Contributions: Dimension 2', ylim = c(-.03, 1.2*max(varCtr2)),threshold = 1 / nrow(varC
print(a0006.Var.ctr2)
```

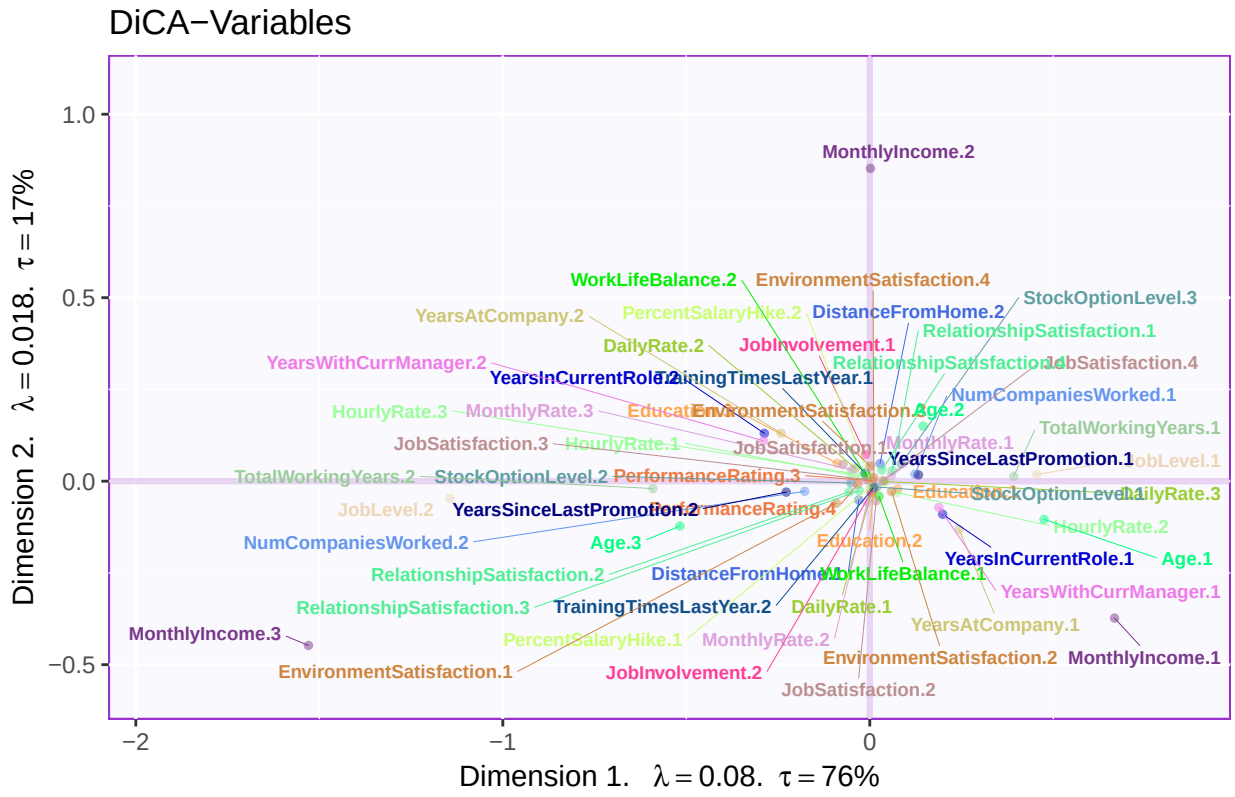



```
varCtr3 <- varCtr[,3]
names(varCtr3) <- rownames(varCtr)
a0006.Var.ctr3 <- PrettyBarPlot2(varCtr3,
main = 'Variable Contributions: Dimension 3', #ylim = c(-.03, 1.2*max(varCtr2)),
threshold = 1 / nrow(varCtr), color4bar = gplots::col2hex(color)
)
print(a0006.Var.ctr3)
```

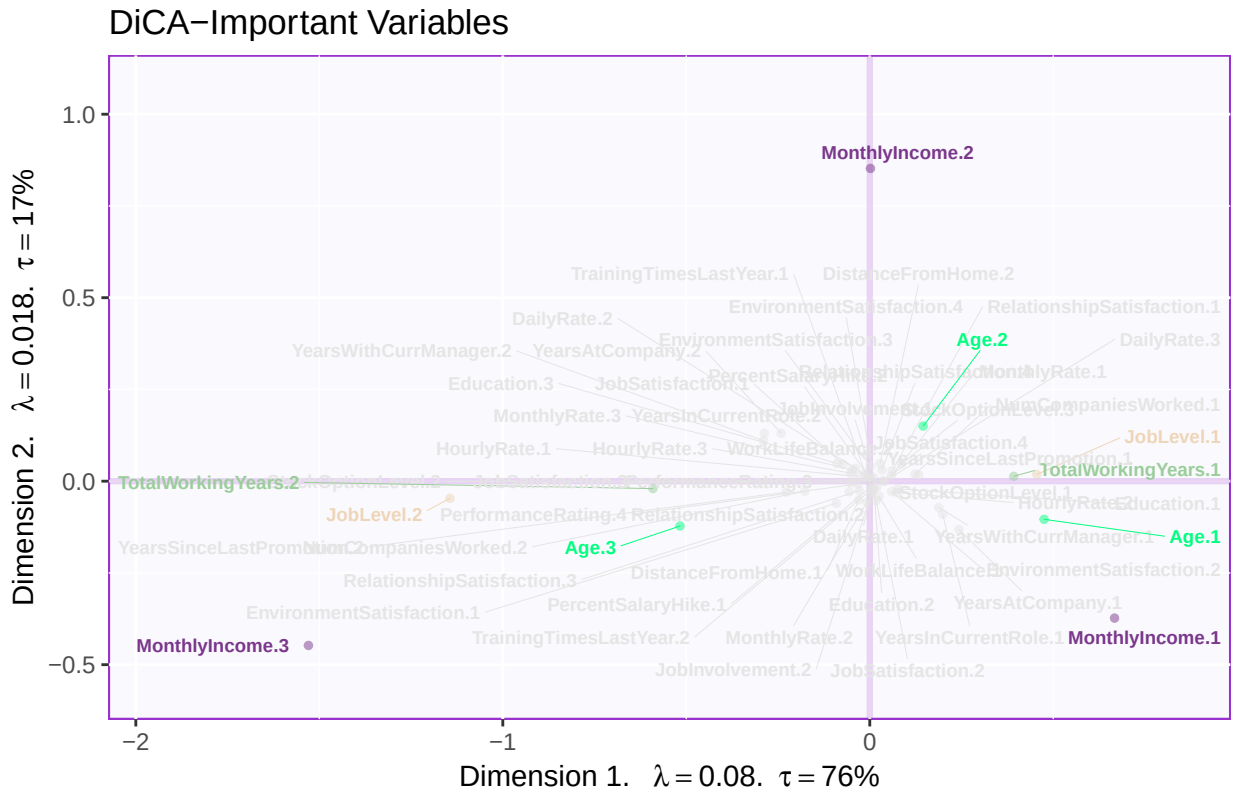


Variable Map

```
axis1 = 1
axis2 = 2
Fj <- resDICA$TExPosition.Data$fj
col4Levels <- data4PCCAR::coloringLevels( rownames(resDICA$TExPosition.Data$fj), color)
col4Labels <- col4Levels$color4Levels

BaseMap.Fj <- createFactorMap(X = Fj , # resMCA$ExPosition.Data$fj,
                              axis1 = axis1, axis2 = axis2,
                              title = 'DiCA-Variables',
                              col.points = col4Labels, cex = 1,
                              col.labels = col4Labels, text.cex = 2.5,
                              force = 2)
labels4MCA <- createxyLabels.gen(x_axis = axis1, y_axis = axis2, lambda = resDICA$TExPosition.Data$eigs
tau = resDICA$TExPosition.Data$t)

b0002.BaseMap.Fj <- BaseMap.Fj$zeMap + labels4MCA
b0003.BaseMapNoDot.Fj <- BaseMap.Fj$zeMap_background +
BaseMap.Fj$zeMap_text + labels4MCA
print(b0002.BaseMap.Fj)
```

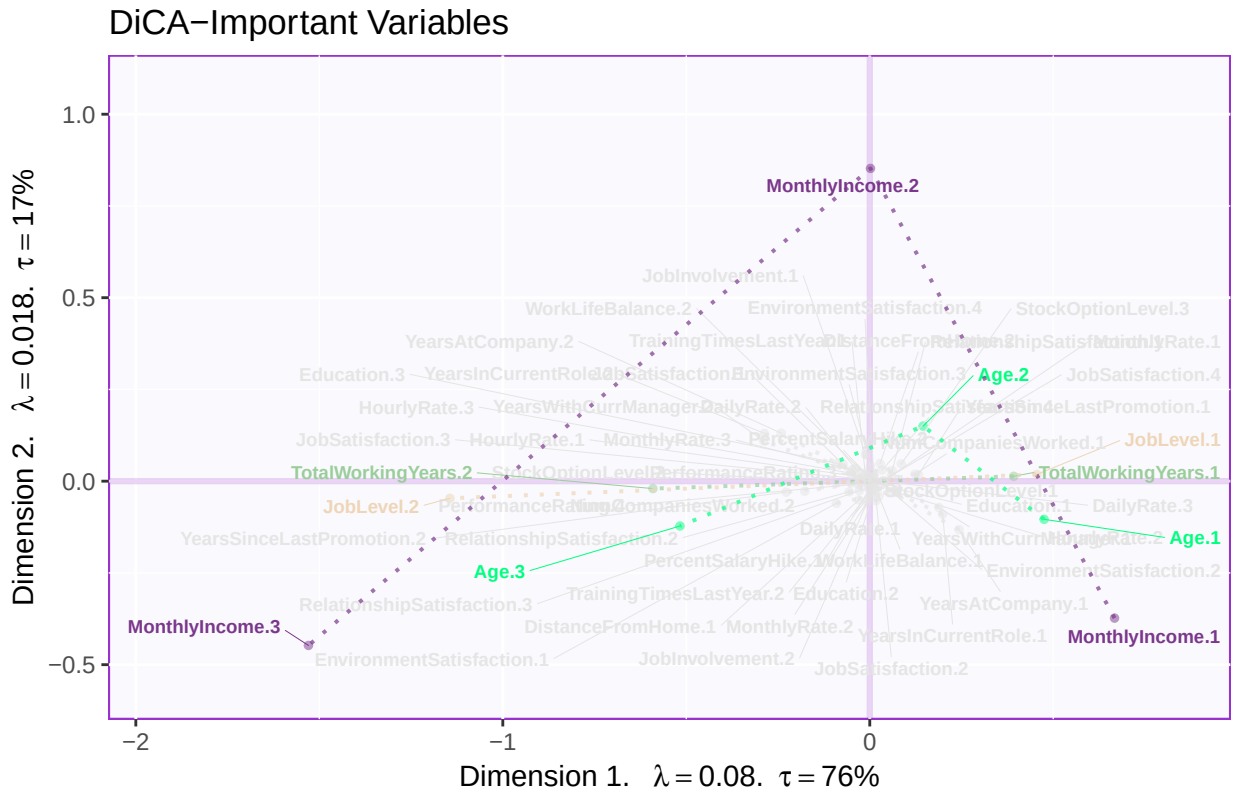



```
absCtrVar <- as.matrix(varCtr) %*% diag(resDICA$TExPosition.Data$eigs)
varCtr12 <- (absCtrVar[,1] + absCtrVar[,2]) /(resDICA$TExPosition.Data$eigs[1] +resDICA$TExPosition.Data
importantVar <- (varCtr12 >= 1 / length(varCtr12))
col4ImportantVar <- color
col4NS <- 'gray90'
col4ImportantVar[!importantVar] <- col4NS
col4Levels.imp <- data4PCCAR::coloringLevels(rownames(Fj), col4ImportantVar)
BaseMap.Fj.imp <- createFactorMap(X = Fj , # resMCA$ExPosition.Data$fj,
                                  axis1 = axis1, axis2 = axis2,
title = 'DiCA-Important Variables', col.points = col4Levels.imp$color4Levels,
cex = 1,
col.labels = col4Levels.imp$color4Levels,
                               text.cex = 2.5,
force = 2)
b0010.BaseMap.Fj <- BaseMap.Fj.imp$zeMap + labels4MCA
print(b0010.BaseMap.Fj)
```



Map with important variables and lines

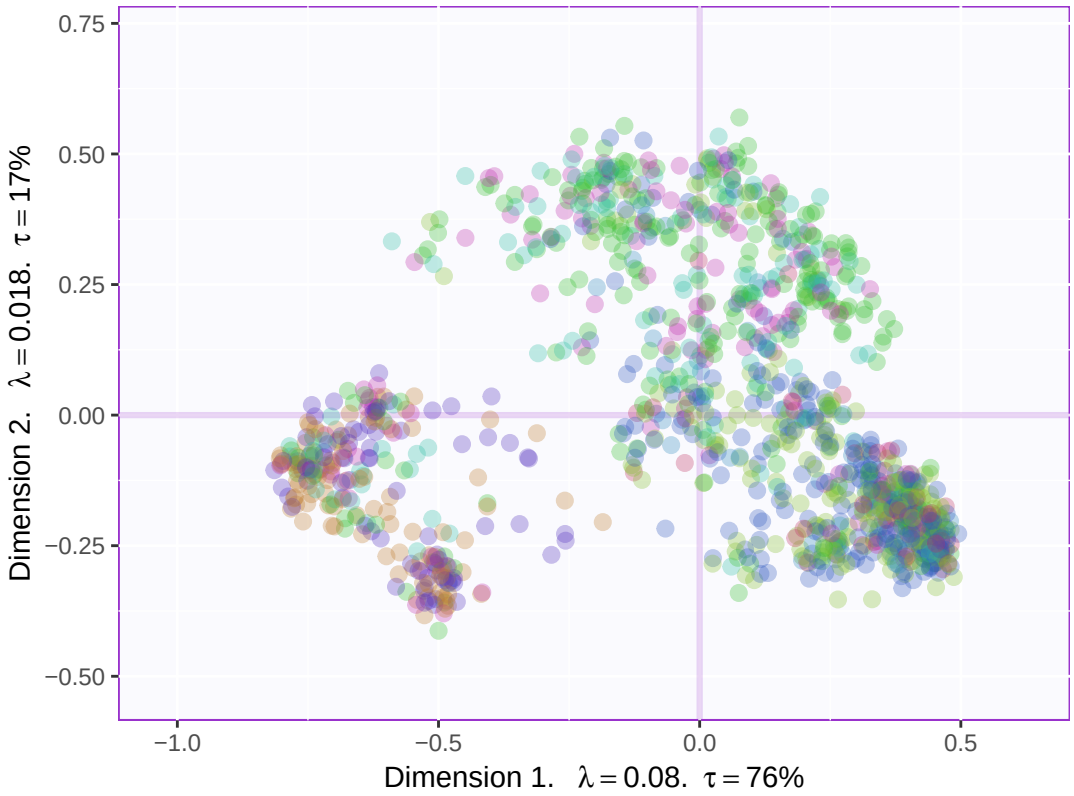
```
lines4J <- addLines4MCA(Fj, col4Var = col4Levels.imp$color4Variables, size = .7)
b0020.BaseMap.Fj <- b0010.BaseMap.Fj + lines4J
print( b0020.BaseMap.Fj)
```



7.3 Factor Map I

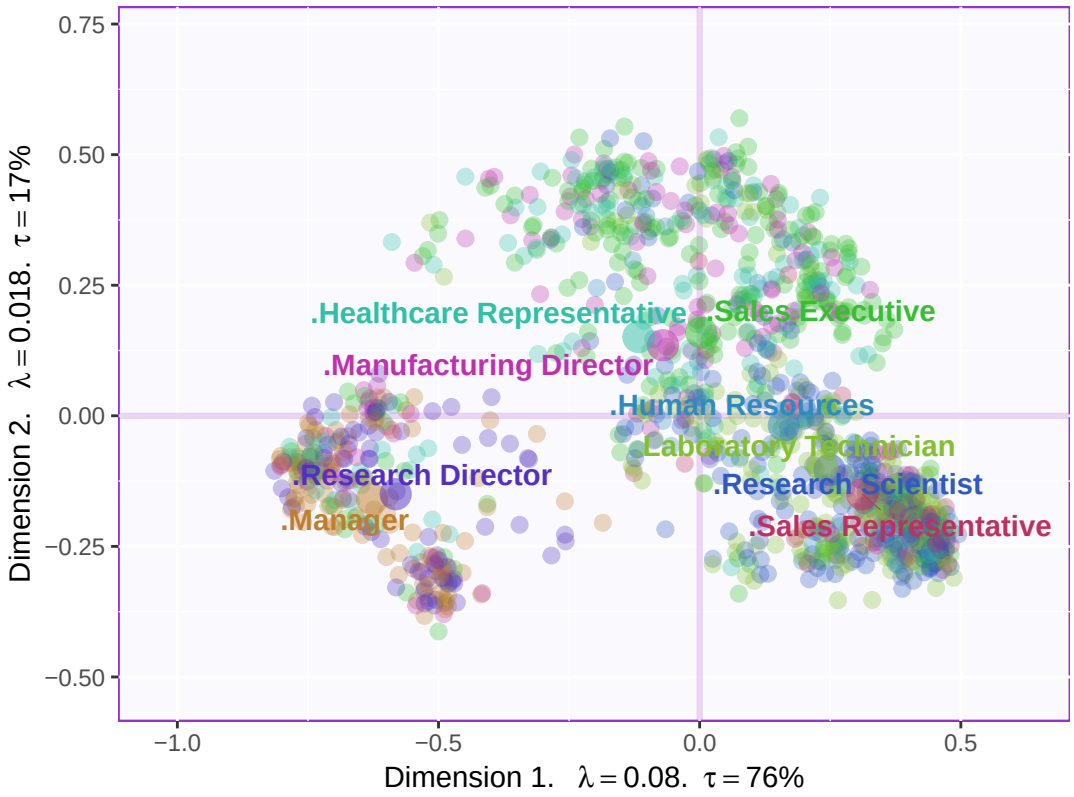
Job Role

```
baseMap.i1 <- PTCA4CATA::createFactorMap(resDICA$TExPosition.Data$fii,
                                         col.points   = resDICA$Plotting.Data$fii.col,
                                         alpha.points =  .3)
label4Map <- createxyLabels.gen(1,2,
                                lambda = resDICA$TExPosition.Data$eigs,
                                tau = resDICA$TExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i1 <- baseMap.i1$zeMap_background + baseMap.i1$zeMap_dots + label4Map
#-----------------------------------------------------------------------
print(aggMap.i1)
```



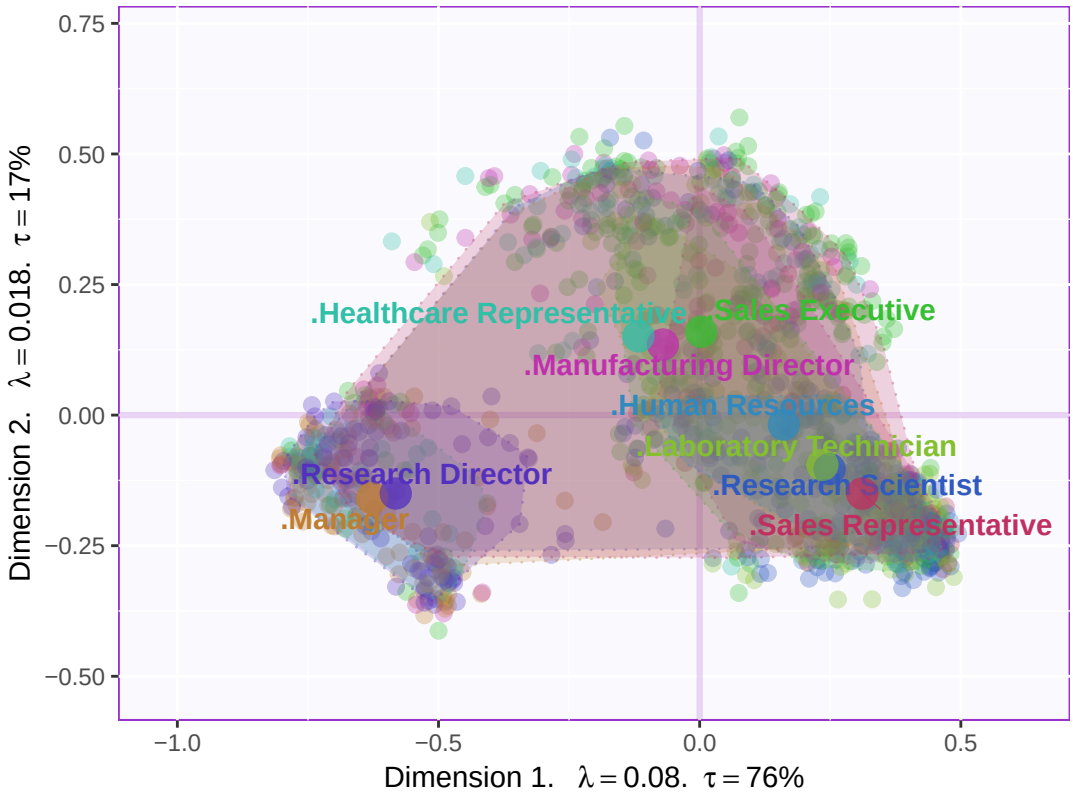

```
col4data1 <- resDICA$Plotting.Data$fii.col
col4Means1 <- unique(col4data1)

MapGroup1    <- PTCA4CATA::createFactorMap(resDICA$TExPosition.Data$fi,
                                axis1 = 1, axis2 = 2,
                                constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means1,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means1,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                             col.background = adjustcolor("lavender",
                                                          alpha.f = 0.2),
                             force = 1, segment.size = 0)

aggMap.i.withMeans1 <- aggMap.i1+
  MapGroup1$zeMap_dots + MapGroup1$zeMap_text
#------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans1)
```



Create 75% Tolerance interval polygons

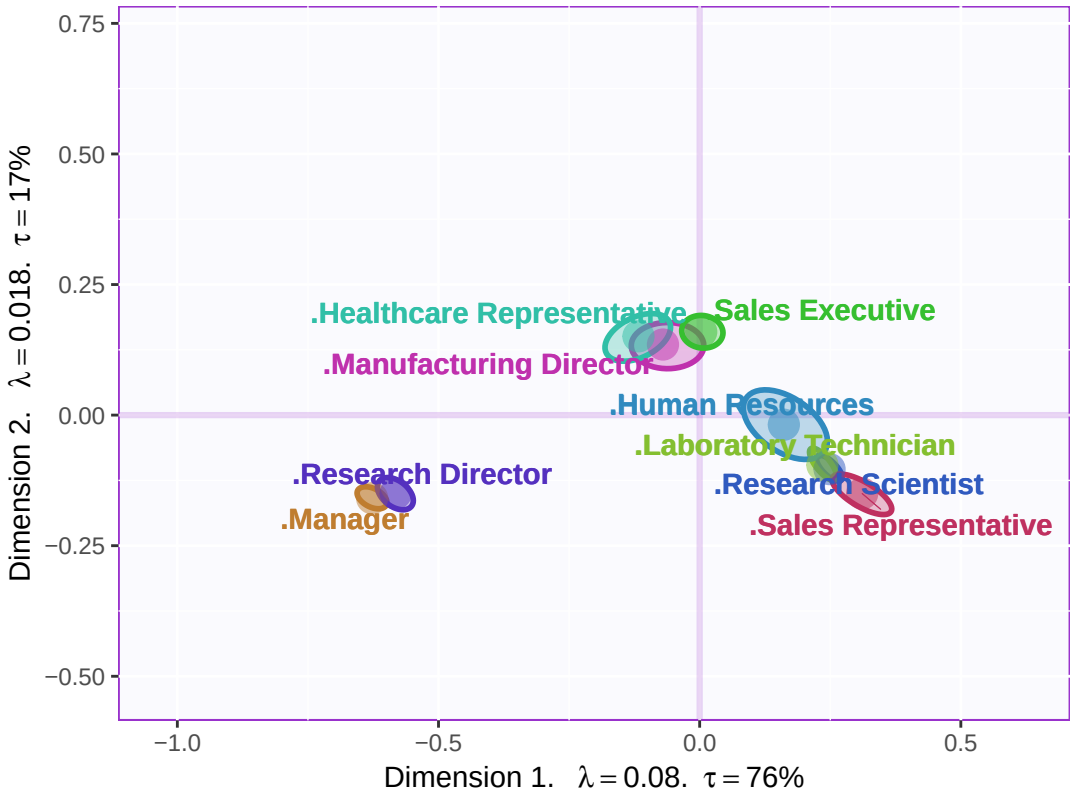
```
GraphTI.Hull.901 <- MakeToleranceIntervals(resDICA$TExPosition.Data$fii,
                                   as.factor(datae),
                                   names.of.factors = c("Dim1","Dim2"),
                                   col = unique(col4data1),
                                   line.size = .5, line.type = 3,
                                   alpha.ellipse = .2,
                                   alpha.line = .4,
                                   p.level = .75, # 75% TI
                                   type = 'hull' #
                                           # use 'hull' for convex hull
)
aggMap.i.withHull1 <- aggMap.i1 +
  GraphTI.Hull.901 + MapGroup1$zeMap_dots +
  MapGroup1$zeMap_text +  MapGroup1$zeMap_dots

print(aggMap.i.withHull1)
```



Inferences

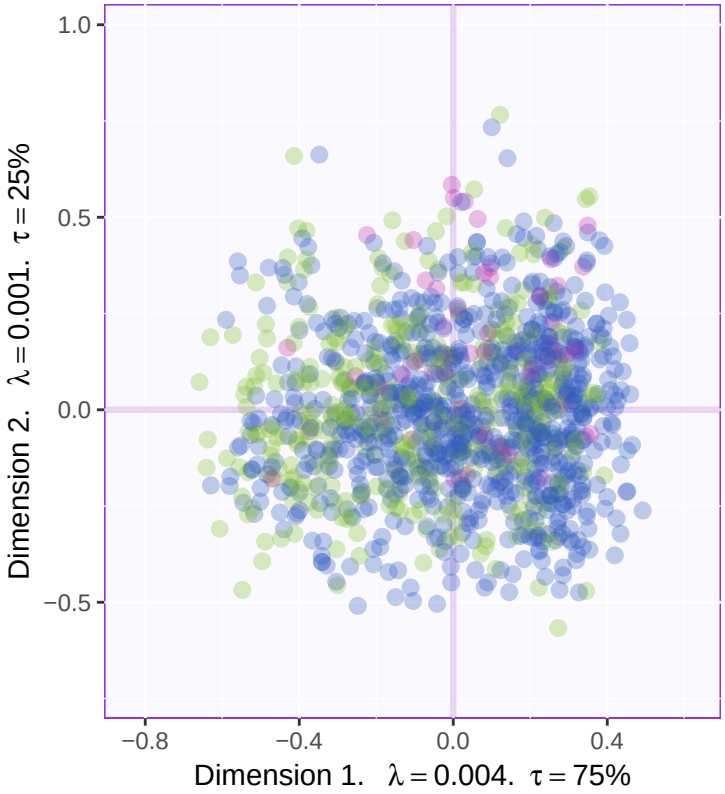
```
fixedCM1   <-   resDICA.inf1$Inference.Data$loo.data$fixed.confuse
looedCM1   <- resDICA.inf1$Inference.Data$loo.data$loo.confuse
```

```
BootCube1 <- resDICA.inf1$Inference.Data$boot.data$fi.boot.data$boots
dimnames(BootCube1)[[2]] <- c("Dimension 1","Dimension 2")

GraphElli1 <- MakeCIEllipses(BootCube1[,1:2,],
                  names.of.factors = c("Dimension 1","Dimension 2"),
                  col = col4Means1,
                  p.level = .95
)


aggMap.i.withCI1 <-  MapGroup1$zeMap_background +  GraphElli1 + MapGroup1$zeMap_text + label4Map + MapG

print(aggMap.i.withCI1)
```



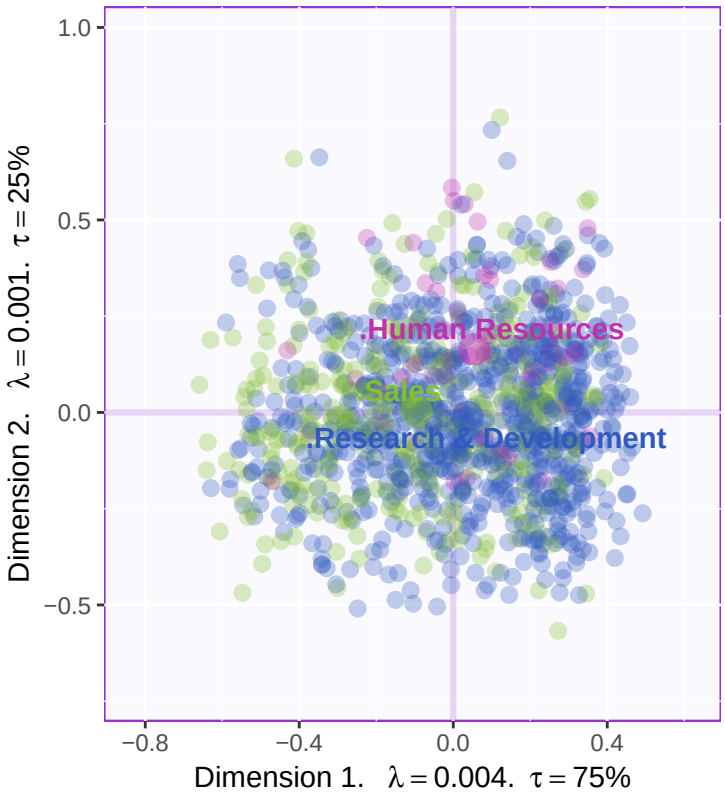
Department

```
baseMap.i11 <- PTCA4CATA::createFactorMap(resDICA1$TExPosition.Data$fii,
                                          col.points   = resDICA1$Plotting.Data$fii.col,
                                          alpha.points =  .3)
label4Map1 <- createxyLabels.gen(1,2,
                                 lambda = resDICA1$TExPosition.Data$eigs,
                                 tau = resDICA1$TExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i11 <- baseMap.i11$zeMap_background + baseMap.i11$zeMap_dots + label4Map1
#-----------------------------------------------------------------------
# print this Map
print(aggMap.i11)
```

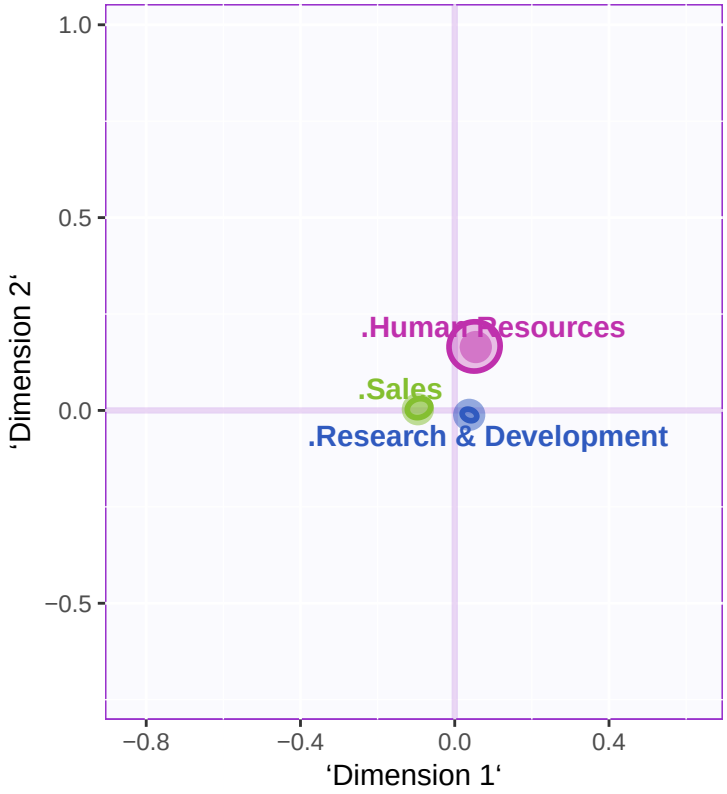



```
col4data <- resDICA1$Plotting.Data$fii.col
col4Means <- unique(col4data)

MapGroup   <- PTCA4CATA::createFactorMap(resDICA1$TExPosition.Data$fi,
                                axis1 = 1, axis2 = 2,
                                constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans <- aggMap.i11+
  MapGroup$zeMap_dots + MapGroup$zeMap_text
#------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans)
```

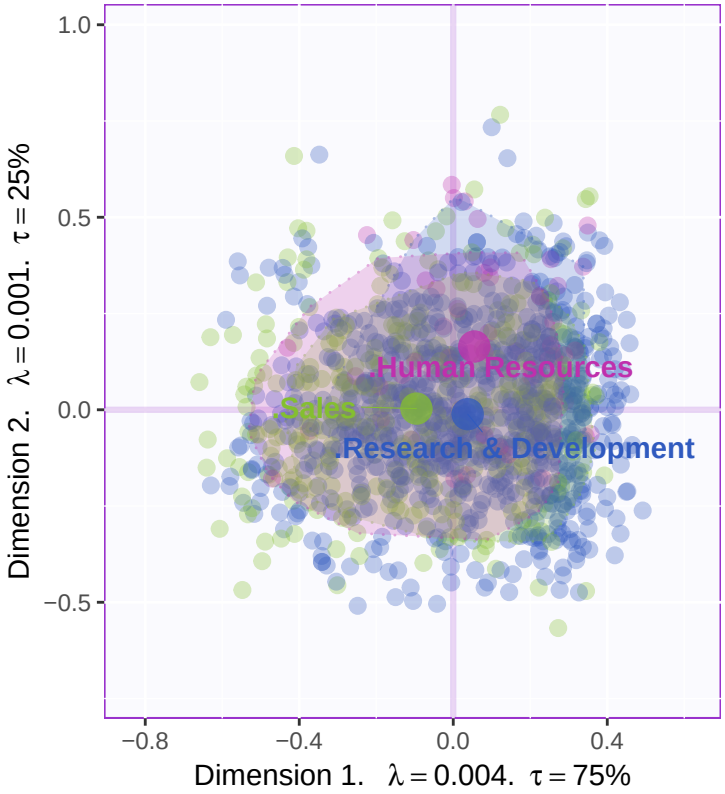



```
resDICA.inf <- tepDICA.inference.battery(DATA = data1,
                    DESIGN = datar,
                    make_design_nominal = TRUE,
                    make_data_nominal = TRUE,
                    group.masses = NULL,
                    weights = NULL,
                    graphs = FALSE,
                    k = 2,
                    test.iters = 100,
                    critical.value = 2)

BootCube <- resDICA.inf$Inference.Data$boot.data$fi.boot.data$boots
dimnames(BootCube)[[2]] <- c("Dimension 1","Dimension 2")

GraphElli <- MakeCIEllipses(BootCube[,1:2,],
                    names.of.factors = c("Dimension 1","Dimension 2"),
                    col = col4Means,
                    p.level = .95
)


aggMap.i.withCI <-  baseMap.i11$zeMap_background +  GraphElli + MapGroup$zeMap_text +   MapGroup$zeMap_

print(aggMap.i.withCI)
```



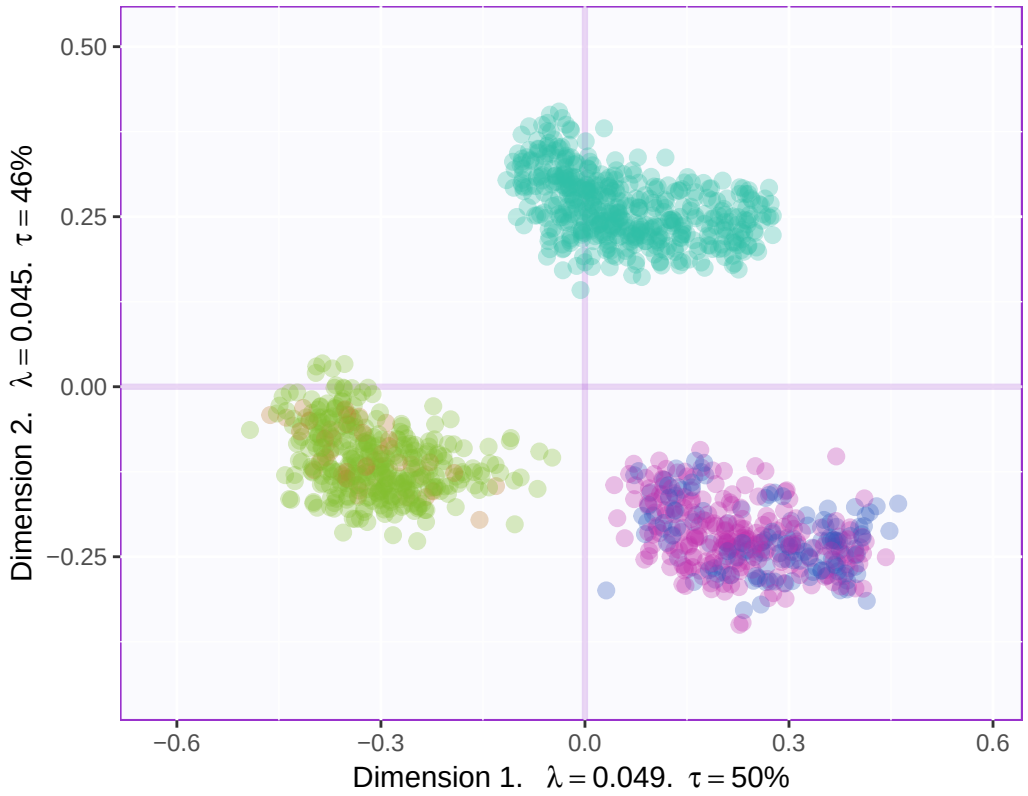
Create 75% Tolerance interval polygons

```
GraphTI.Hull <- MakeToleranceIntervals(resDICA1$TExPosition.Data$fii,
                                       as.factor(datar),
                                       names.of.factors = c("Dim1","Dim2"),
                                       col = unique(col4data),
                                       line.size = .5, line.type = 3,
                                       alpha.ellipse = .2,
                                       alpha.line = .4,
                                       p.level = .75, # 75% TI
                                       type = 'hull' #
                                              # use 'hull' for convex hull
)
aggMap.i.withHull <- aggMap.i11 +
  GraphTI.Hull + MapGroup$zeMap_dots +
  MapGroup$zeMap_text +  MapGroup$zeMap_dots + label4Map1

print(aggMap.i.withHull)
```



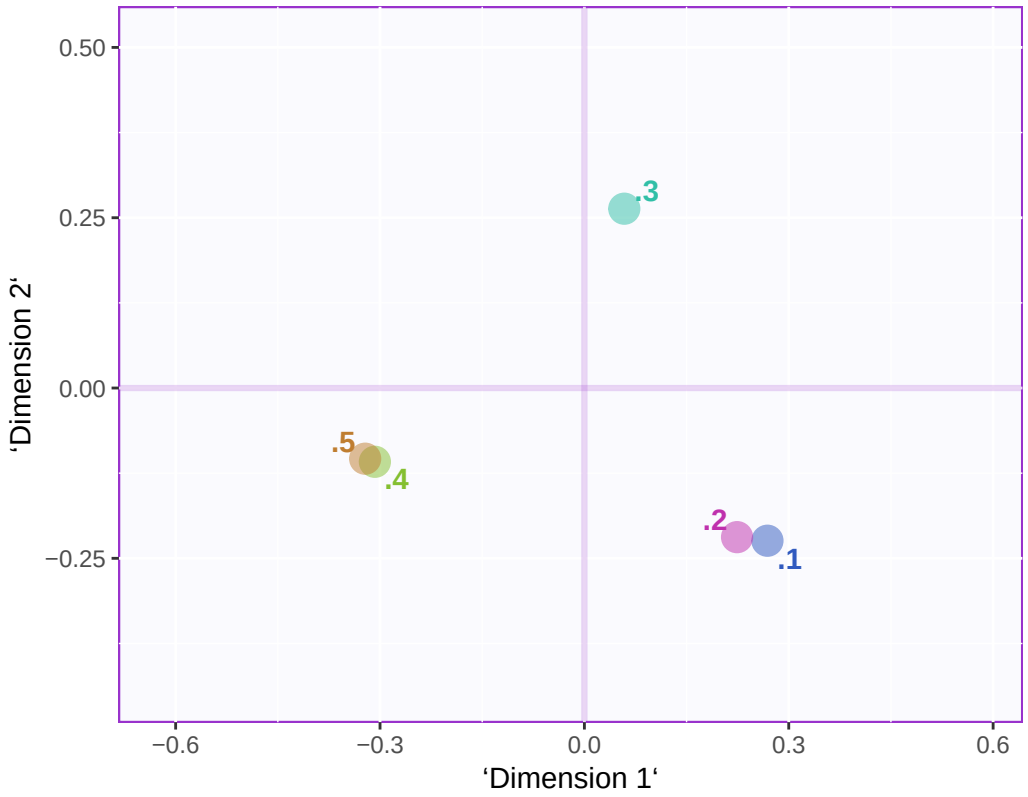
Education

```
baseMap.i112 <- PTCA4CATA::createFactorMap(resDICA2$TExPosition.Data$fii,
                                           col.points    = resDICA2$Plotting.Data$fii.col,
                                           alpha.points =  .3)
label4Map12 <- createxyLabels.gen(1,2,
                                  lambda = resDICA2$TExPosition.Data$eigs,
                                  tau = resDICA2$TExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i112 <- baseMap.i112$zeMap_background + baseMap.i112$zeMap_dots + label4Map12
#-------------------------------------------------------------------------
# print this Map
print(aggMap.i112)
```

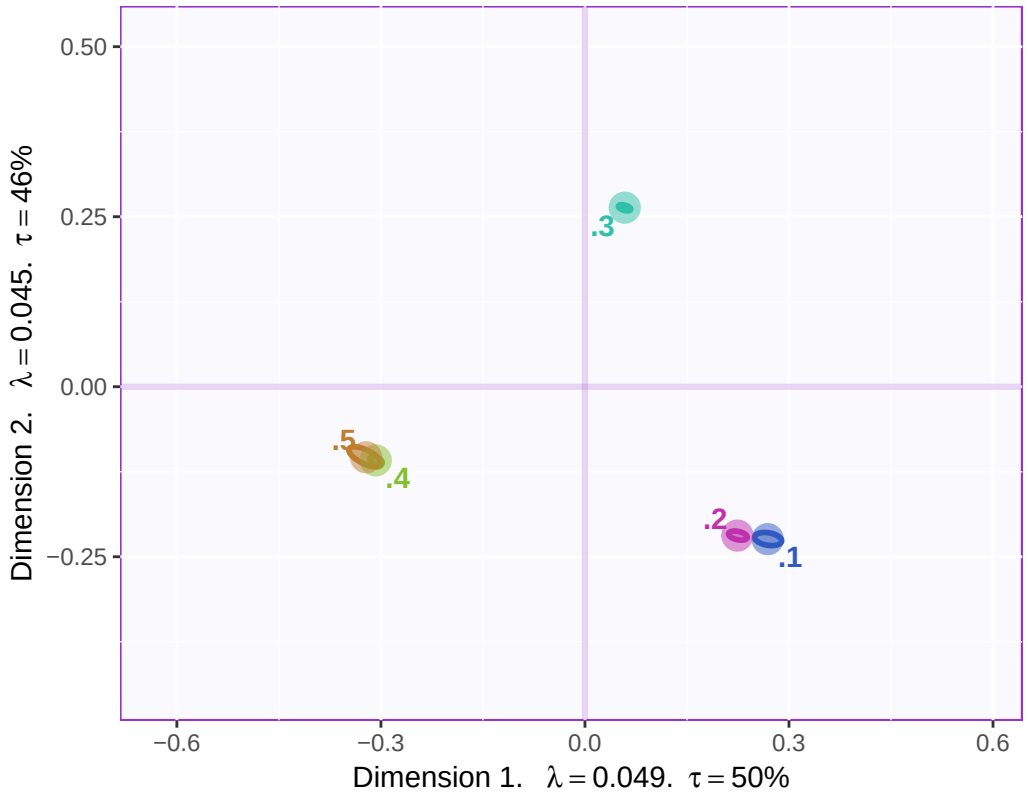



```
col4data2 <- resDICA2$Plotting.Data$fii.col
col4Means2 <- unique(col4data2)

MapGroup2   <- PTCA4CATA::createFactorMap(resDICA2$TExPosition.Data$fi,
                                axis1 = 1, axis2 = 2,
                                constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means2,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means2,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans2 <- baseMap.i112$zeMap_background +
  MapGroup2$zeMap_dots + MapGroup2$zeMap_text
#------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans2)
```

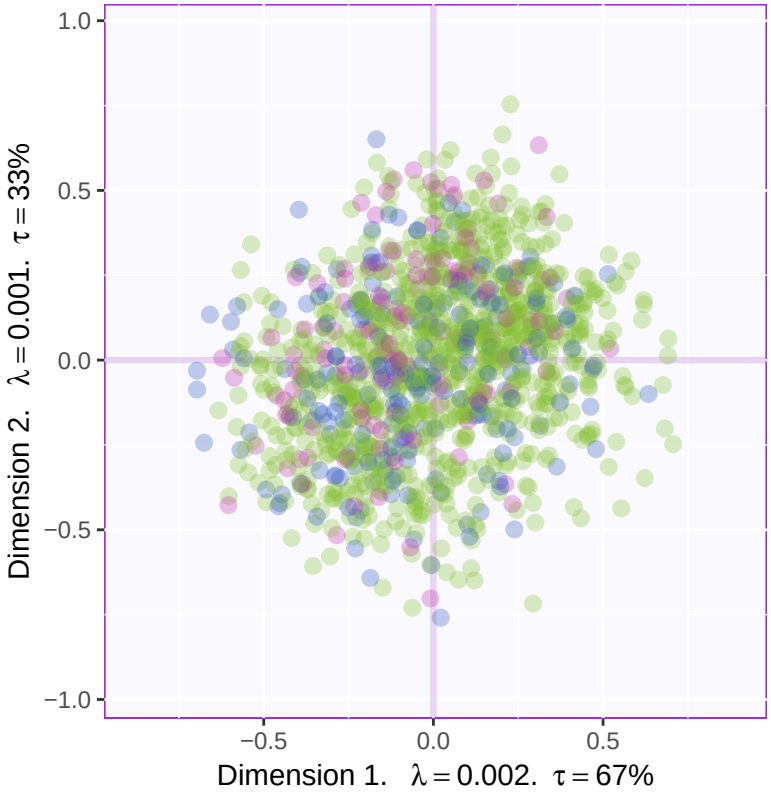



```
BootCube2 <- resDICA.inf2$Inference.Data$boot.data$fi.boot.data$boots
dimnames(BootCube2)[[2]] <- c("Dimension 1","Dimension 2")

GraphElli2 <- MakeCIEllipses(BootCube2[,1:2,],
                  names.of.factors = c("Dimension 1","Dimension 2"),
                  col = col4Means2,
                  p.level = .95
)


aggMap.i.withCI2 <- baseMap.i112$zeMap_background +  GraphElli2 + MapGroup2$zeMap_text + label4Map12 + l

print(aggMap.i.withCI2)
```



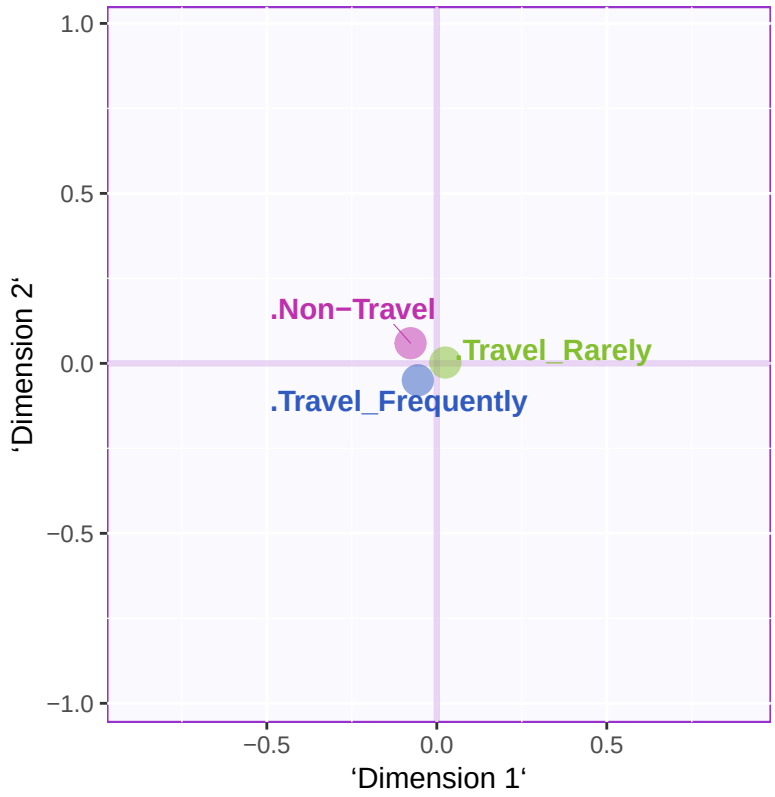
Business Travel

```
baseMap.i1123 <- PTCA4CATA::createFactorMap(resDICA3$TExPosition.Data$fii,
                                            col.points    = resDICA3$Plotting.Data$fii.col,
                                            alpha.points =  .3)
label4Map123 <- createxyLabels.gen(1,2,
                                   lambda = resDICA3$TExPosition.Data$eigs,
                                   tau = resDICA3$TExPosition.Data$t )

# Plain map with color for the I-set
aggMap.i1123 <- baseMap.i1123$zeMap_background + baseMap.i1123$zeMap_dots + label4Map123
#-----------------------------------------------------------------------
# print this Map
print(aggMap.i1123)
```

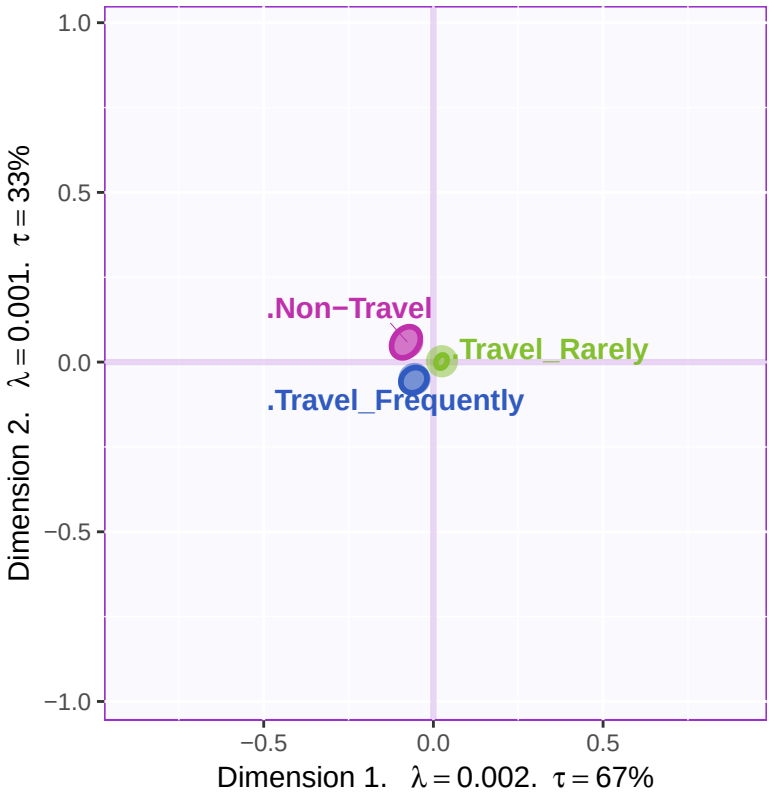



```
col4data3 <- resDICA3$Plotting.Data$fii.col
col4Means3 <- unique(col4data3)

MapGroup3   <- PTCA4CATA::createFactorMap(resDICA3$TExPosition.Data$fi,
                                axis1 = 1, axis2 = 2,
                                constraints = baseMap.i1$constraints,
                                title = NULL,
                                col.points = col4Means3,
                                display.points = TRUE,
                                pch = 19, cex = 5,
                                display.labels = TRUE,
                                col.labels = col4Means3,
                                text.cex = 4,
                                font.face = "bold",
                                font.family = "sans",
                                col.axes = "darkorchid",
                                alpha.axes = 0.2,
                                width.axes = 1.1,
                            col.background = adjustcolor("lavender",
                                                         alpha.f = 0.2),
                            force = 1, segment.size = 0)

aggMap.i.withMeans23 <- baseMap.i1123$zeMap_background +
  MapGroup3$zeMap_dots + MapGroup3$zeMap_text
#------------------------------------------------------------------------
```

```
print(aggMap.i.withMeans23)
```

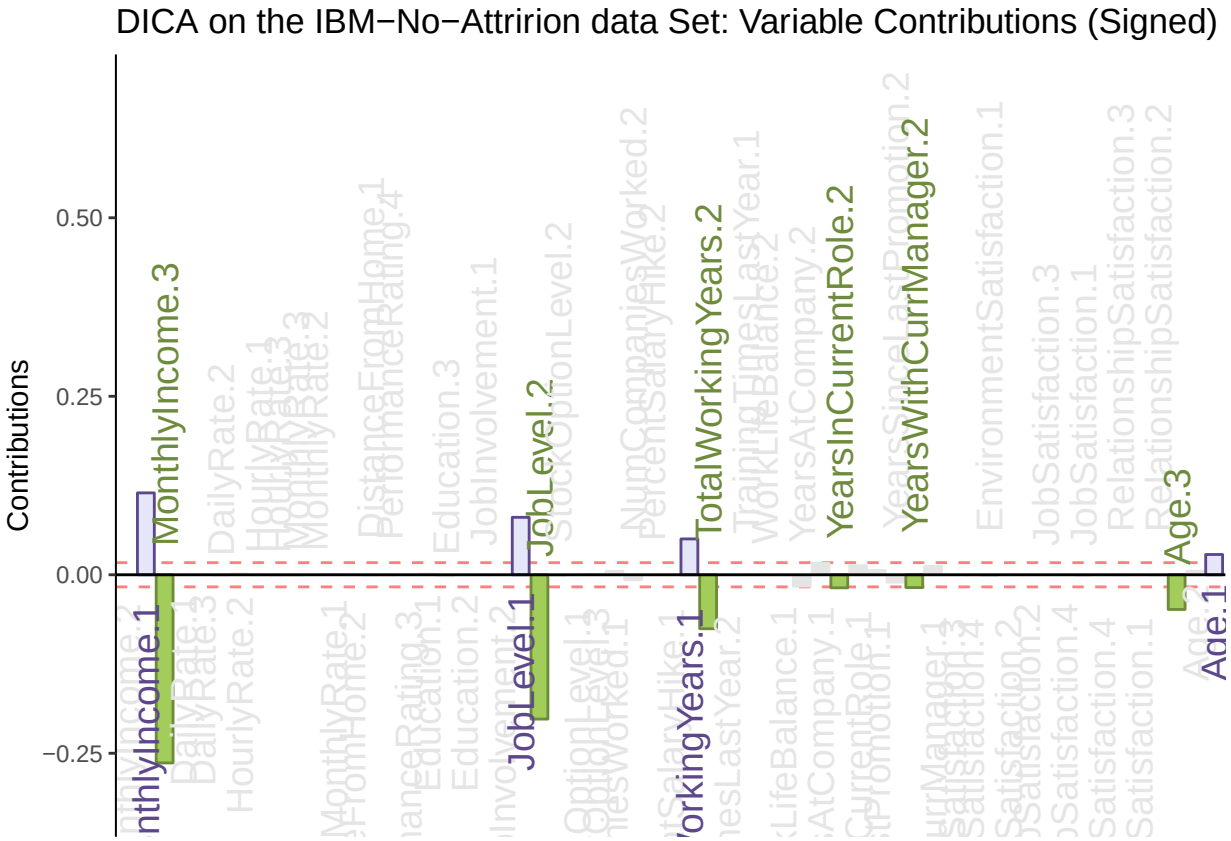



```
BootCube23 <- resDICA.inf3$Inference.Data$boot.data$fi.boot.data$boots
dimnames(BootCube23)[[2]] <- c("Dimension 1","Dimension 2")

GraphElli3 <- MakeCIEllipses(BootCube23[,1:2,],
                  names.of.factors = c("Dimension 1","Dimension 2"),
                  col = col4Means3,
                  p.level = .95
)


aggMap.i.withCI23 <-   baseMap.i1123$zeMap_background +  GraphElli3 + MapGroup3$zeMap_text + label4Map1

print(aggMap.i.withCI23)
```

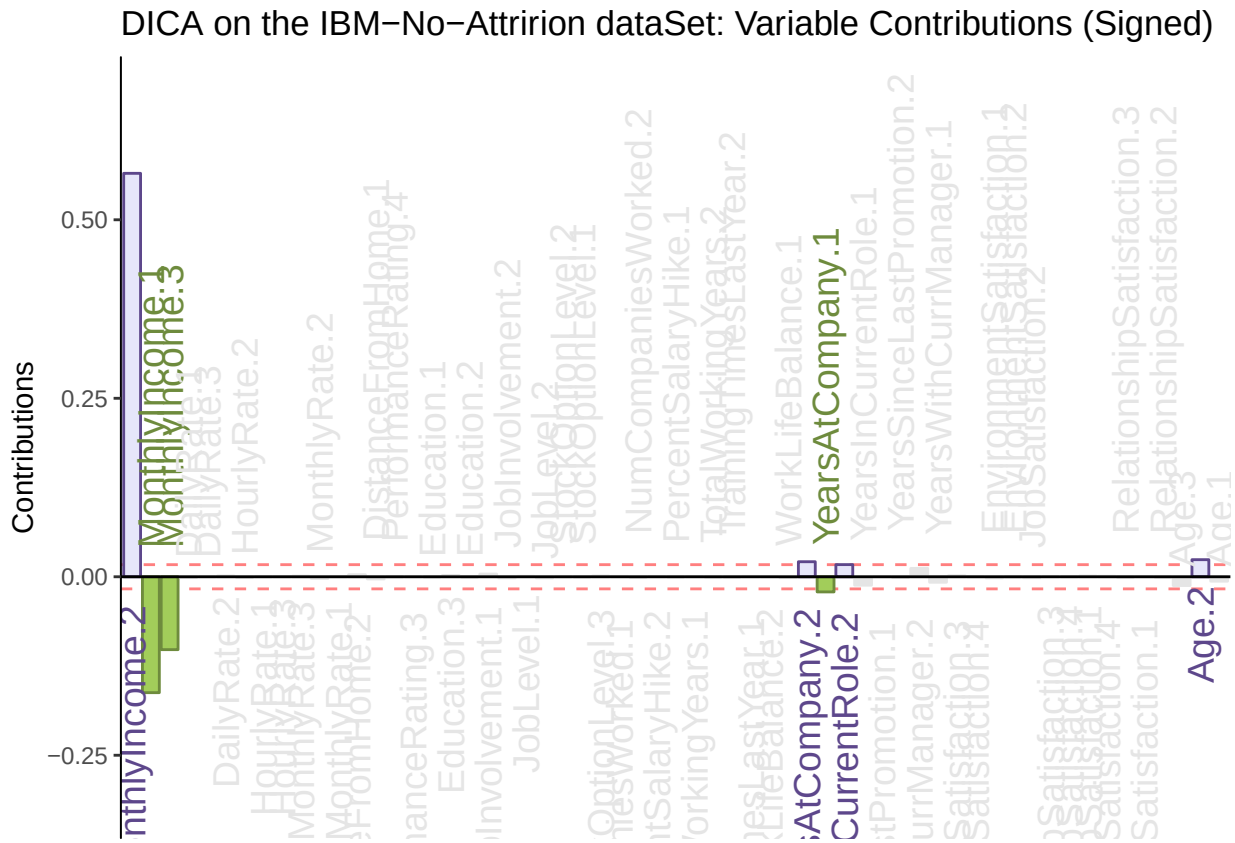


7.4 Contribution

```
signed.ctrJ <- resDICA$TExPosition.Data$cj * sign(resDICA$TExPosition.Data$fj)
b003.ctrJ.s.1 <- PrettyBarPlot2(signed.ctrJ[,1],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                                # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'DICA on the IBM-No-Attririon data Set: Variable Contributions (S
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ))
)
print(b003.ctrJ.s.1)
```




```
b004.ctrJ.s.2 <- PrettyBarPlot2(signed.ctrJ[,2],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                                # color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'DICA on the IBM-No-Attririon dataSet: Variable Contributions (S
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ))
)
print(b004.ctrJ.s.2)
```



7.5 Bootstrap Ratios

```
BR <- resDICA.inf1$Inference.Data$boot.data$fj.boot.data$tests$boot.ratios
laDim = 1
ba001.BR1 <- PrettyBarPlot2(BR[,laDim],
                            threshold = 2,
                            font.size = 5,
                            #color4bar = gplots::col2hex(col4J.ibm),
                            main = paste0( 'DICA on the IBM-NoAttrition data Set: Bootstrap ratio ',laD
                            ylab = 'Bootstrap ratios'
                            #ylim = c(1.2*min(BR[,laDim]), 1.2*max(BR[,laDim]))
)
print(ba001.BR1)
```


```
#
laDim = 2
ba002.BR2 <- PrettyBarPlot2(BR[,laDim],
                            threshold = 2,
                            font.size = 5,
                            #color4bar = gplots::col2hex(col4J.ibm),
                            main = paste0(
                              'DICA on the IBM-NoAttrition data Set: Bootstrap ratio ',laDim),
                            ylab = 'Bootstrap ratios'
)
print(ba002.BR2)
```

7.6 Summary

Since, each observation is assigned to the closest group in DiCA the results infered here are pretty significant and apart from each other so we can confidently vouch for the difference in groups.DiCA is intrepreted as CA, we using distance metric we can give the following conclusions :

Component 1: - We can say that the Research Director, Manager, Research Scientists have high Income and higher experience in comparison to other Job Roles. Also, we can say that the Sales People have least Income and less number of experience.

- Generally, Managers and Directors are the ones with PhD & Master's with higher job level, Sales representative and Lab Technician have no college education while Healthcare Representatives & Manufacturing Directors have Bachelor's degree
- People who are Managers and Research Directors have the highest pay and are more seasoned while people who are Sales representative and Lab Technician have low pay and less experience
- It is also observed that the Mangers and Research Directors travel the most, could be their meetings and conferences while the HR, Sales representative,Lab Technician, Research Scientists travel occasionally
- Also, we can say here that Age is also an important factor in the JOb level i.e Lower the position younger the age and higher the position the more seasoned the employees are.

Component 2: - We can also say that component 2 divides HR vs Sales and Research & Development telling that the HR people are generally older with higher income.

```
rm(list = ls())
graphics.off()
```

8 MFA

```
data <- read.csv("IBM-HR-Emplyee-NoAttrition.csv")
data1 <- data[,c(11:29)]
design <- data$JobRole
```

```
a <- as.matrix(c("1","1","1","1","1","2","2","2","2","2","2","2","3","3","3","3","3","3","3"))
a <- t(a)
colnames(a) <- colnames(data1)
resMFA <- mpMFA(data1 ,column.design = a,graphs = FALSE,DESIGN = data$JobRole)
```

```
## [1] "Preprocessed the Rows of the data matrix using:  None"
## [1] "Preprocessed the Columns of the data matrix using:  Center_1Norm"
## [1] "Preprocessed the Tables of the data matrix using:  MFA_Normalization"
## [1] "Preprocessing Completed"
## [1] "Optimizing using:  None"
```

```
## Warning in sqrt(eigOut$values): NaNs produced
```

```
## [1] "Processing Complete"
```

```
resMFA1 <- mpMFA(data1 ,column.design = a,graphs = FALSE,DESIGN = data$Gender)
```

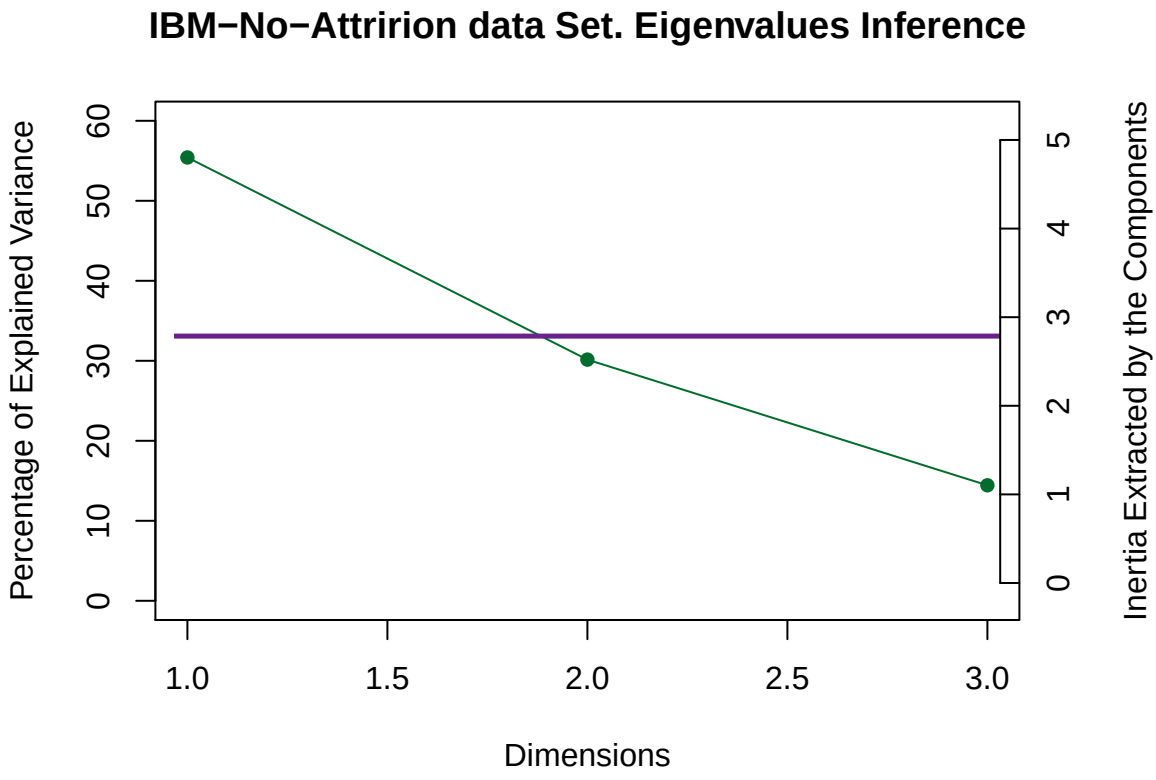
```
## [1] "Preprocessed the Rows of the data matrix using:  None"
## [1] "Preprocessed the Columns of the data matrix using:  Center_1Norm"
## [1] "Preprocessed the Tables of the data matrix using:  MFA_Normalization"
## [1] "Preprocessing Completed"
## [1] "Optimizing using:  None"
```

```
## Warning in sqrt(eigOut$values): NaNs produced
```

```
## [1] "Processing Complete"
```

8.1 PlotScree

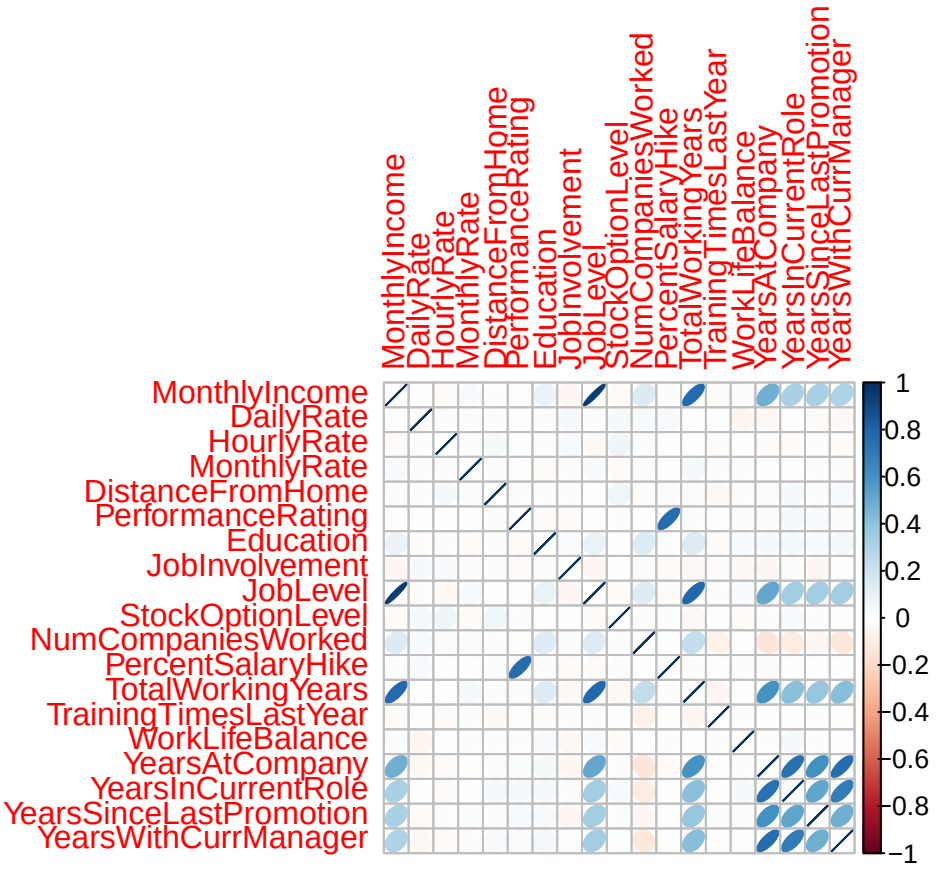
```
PlotScree(ev = resMFA$mexPosition.Data$InnerProduct$eigs,
          p.ev = NULL,
          title = 'IBM-No-Attririon data Set. Eigenvalues Inference',
          plotKaiser = TRUE
)
```




```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

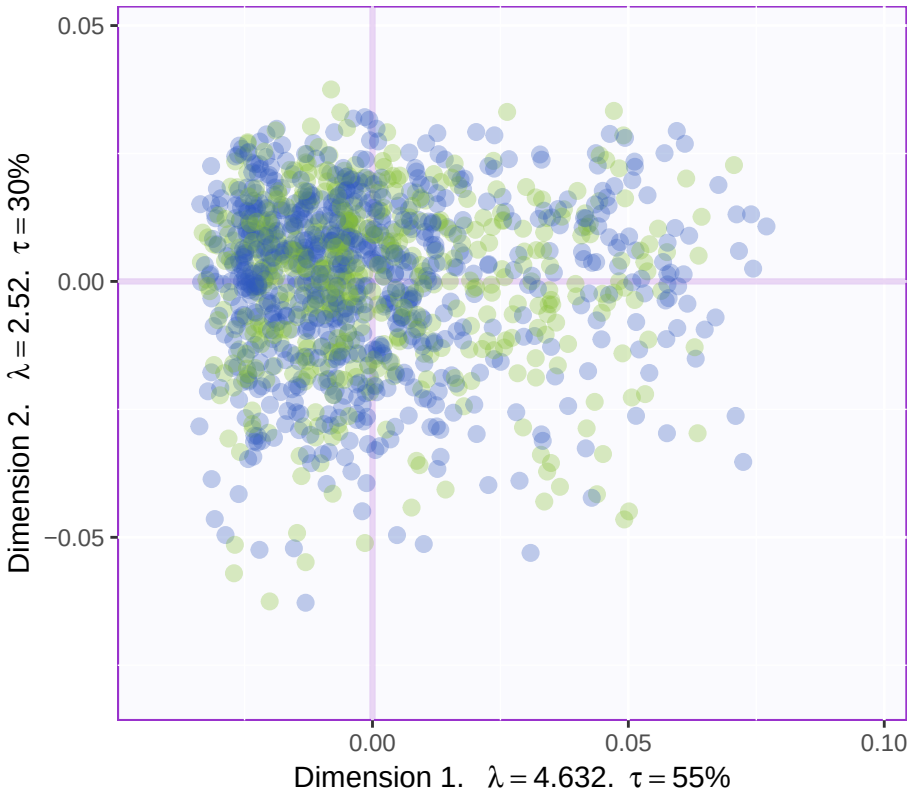
```
cor.my_data <- cor(data1)
corrplot(cor.my_data, method = "ellipse")
```



8.2 Factor Scores

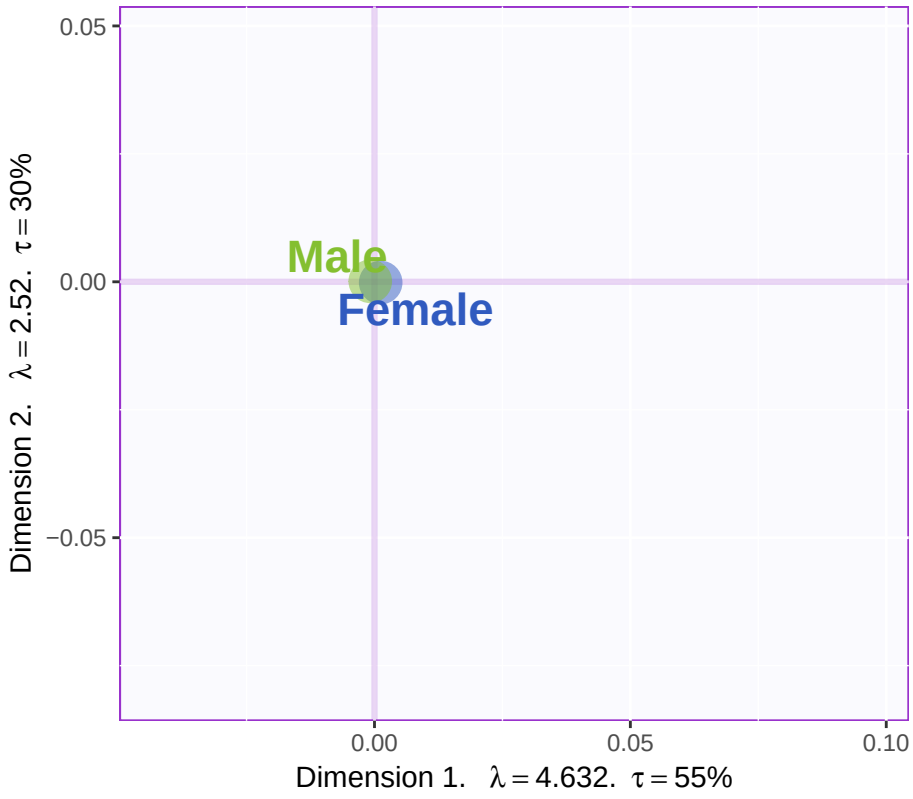
Gender

```
baseMap.j0 <- PTCA4CATA::createFactorMap(resMFA1$mexPosition.Data$Table$fi,col.points = resMFA1$Plottin
                                         col.labels = resMFA$Plotting.Data$fi.col,
                                         alpha.points =  .3,display.labels = FALSE)

label4Map0 <- createxyLabels.gen(1,2,
                                 lambda = resMFA1$mexPosition.Data$InnerProduct$eigs,
                                 tau = resMFA1$mexPosition.Data$InnerProduct$t )
# A graph for the J-set
b000.aggMap.j0 <- baseMap.j0$zeMap_background + # background layer
  baseMap.j0$zeMap_dots + baseMap.j0$zeMap_text + label4Map0
# We print this Map with the following code
print(b000.aggMap.j0)
```



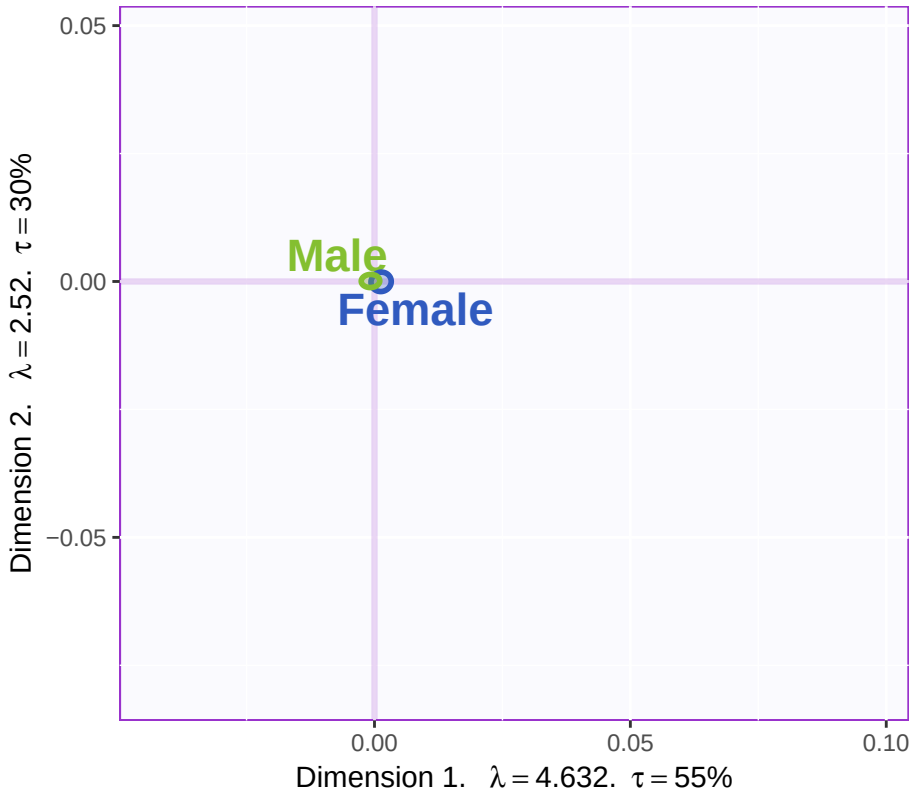

```
# Bootstrap for CI:
BootCube.Gr0 <- PTCA4CATA::Boot4Mean(resMFA1$mexPosition.Data$Table$fi,
                                     design = data$Gender,
                                     niter = 100,
                                     suppressProgressBar = TRUE)
#_____________________________________________________________________
# Bootstrap ratios ----
bootRatios.Gr0 <- boot.ratio.test(BootCube.Gr0$BootCube)
#**********************************************************************
# eigenvalues: MonteCarlo Approach ----
#
random.eigen0 <- data4PCCAR::monteCarlo.eigen(X = data1, nIter = 100)
#
# eigenvalues: Bootstrap approach
#
bootstrap.eigen0 <- data4PCCAR::boot.eigen(data1, nIter = 100)
# Mean Map
#  create the map for the means
#  get the means by groups

dataMeans0 <- PTCA4CATA::getMeans(resMFA1$mexPosition.Data$Table$fi, data$Gender)
# a vector of color for the means
col4data0 <- resMFA1$Plotting.Data$fi.col
col4Means0 <- unique(col4data0)
# the map
MapGroup0 <- PTCA4CATA::createFactorMap(dataMeans0,
```

```
                                    # use the constraint from the main map
                                    constraints = resMFA1$Plotting.Data$constraints,
                                    col.points = col4Means0,
                                    cex = 7,  # size of the dot (bigger)
                                    col.labels = col4Means0,
                                    text.cex = 6)
# The map with observations and group means
a003.Map.I.withMeans0 <-  baseMap.j0$zeMap_background +
  MapGroup0$zeMap_dots + MapGroup0$zeMap_text + label4Map0
print(a003.Map.I.withMeans0)
```



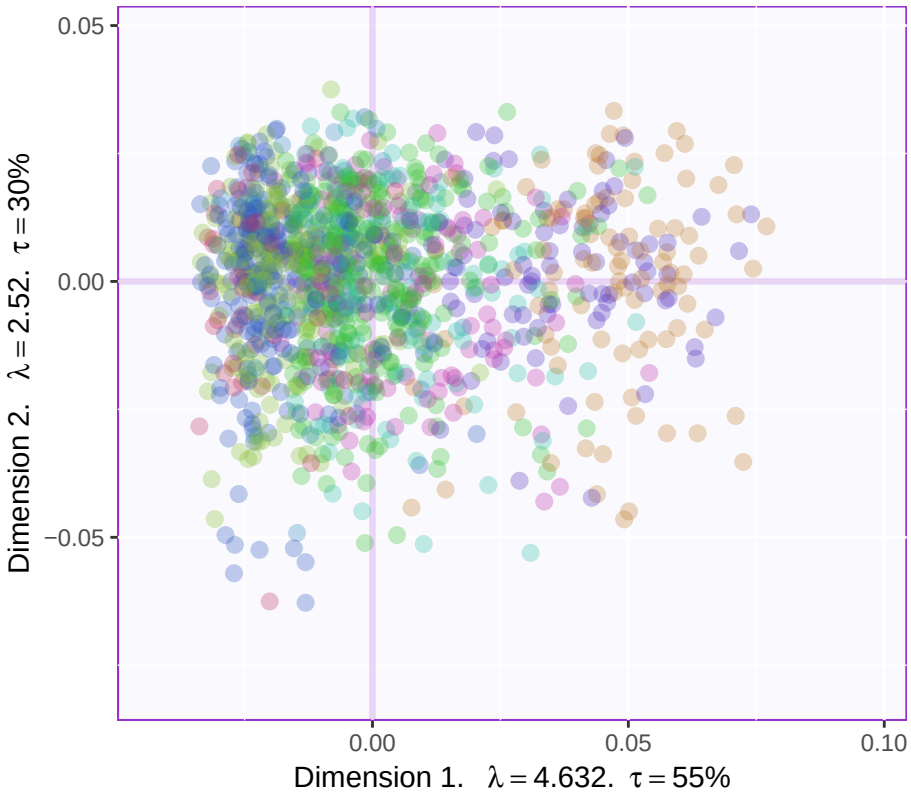

```
#_____________________________________________________________________
# Create the ellipses
# Bootstrapped CI ----
#_____________________________________________________________________
# Create Confidence Interval Plots
# use function MakeCIEllipses from package PTCA4CATA
GraphElli0 <- PTCA4CATA::MakeCIEllipses(BootCube.Gr0$BootCube[,1:2,],
                                    names.of.factors = c("Dimension 1","Dimension 2"),
                                    col = col4Means0,
                                    p.level = .95
)
#_____________________________________________________________________
# create the I-map with Observations, means and confidence intervals
#
a004.Map.I.withCI0 <-  baseMap.j0$zeMap_background + MapGroup0$zeMap_text +  GraphElli0 + label4Map0
```

```
#_____________________________________________________________________
# plot it!
print(a004.Map.I.withCI0)
```



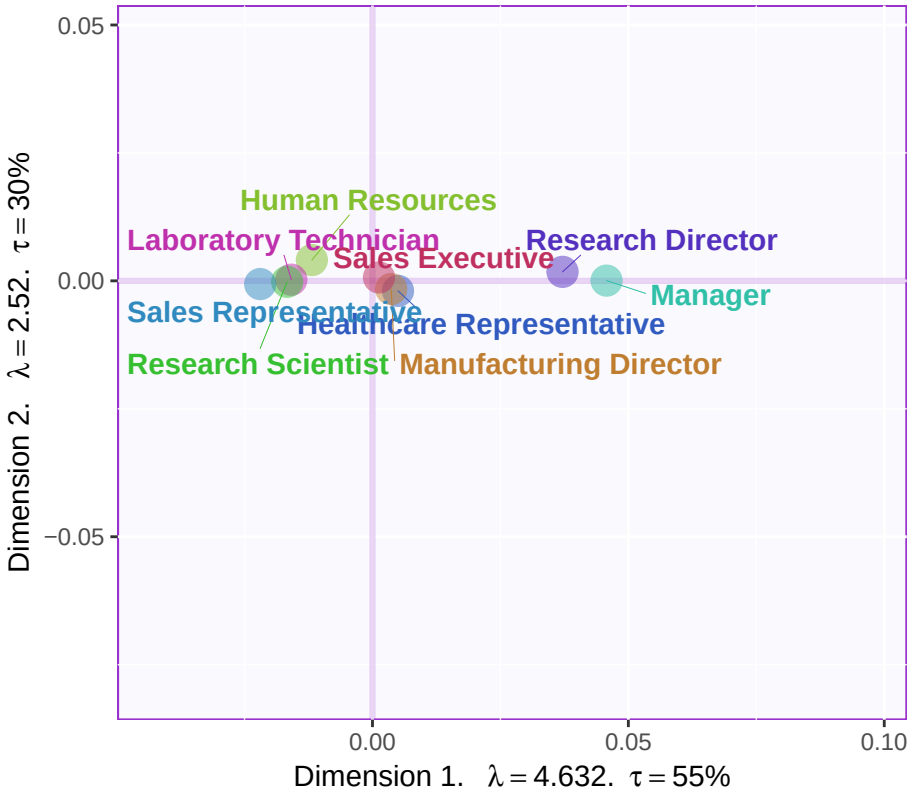
Job Role

```
baseMap.j <- PTCA4CATA::createFactorMap(resMFA$mexPosition.Data$Table$fi,col.points = resMFA$Plotting.Da
                                        col.labels = resMFA$Plotting.Data$fi.col,
                                        alpha.points =  .3,display.labels = FALSE)

label4Map <- createxyLabels.gen(1,2,
                                lambda = resMFA$mexPosition.Data$InnerProduct$eigs,
                                tau = resMFA$mexPosition.Data$InnerProduct$t )
# A graph for the J-set
b000.aggMap.j <- baseMap.j$zeMap_background + # background layer
  baseMap.j$zeMap_dots + baseMap.j$zeMap_text + label4Map
# We print this Map with the following code
print(b000.aggMap.j)
```



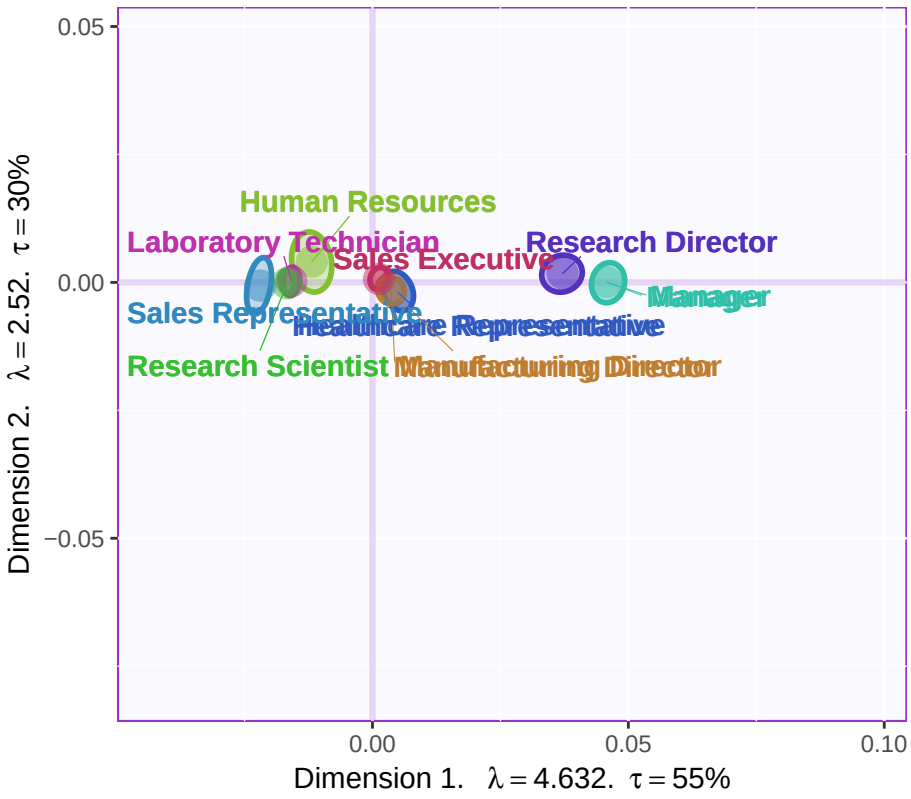

```
# Bootstrap for CI:
BootCube.Gr <- PTCA4CATA::Boot4Mean(resMFA$mexPosition.Data$Table$fi,
                                    design = data$JobRole,
                                    niter = 100,
                                    suppressProgressBar = TRUE)
#_____________________________________________________________________
# Bootstrap ratios ----
bootRatios.Gr <- boot.ratio.test(BootCube.Gr$BootCube)
#**********************************************************************
# eigenvalues: MonteCarlo Approach ----
#
random.eigen <- data4PCCAR::monteCarlo.eigen(X = data1, nIter = 100)
#
# eigenvalues: Bootstrap approach
#
bootstrap.eigen <- data4PCCAR::boot.eigen(data1, nIter = 100)
# Mean Map
#  create the map for the means
#  get the means by groups

dataMeans <- PTCA4CATA::getMeans(resMFA$mexPosition.Data$Table$fi, data$JobRole)
# a vector of color for the means
col4data <- resMFA$Plotting.Data$fi.col
col4Means <- unique(col4data)
# the map
MapGroup <- PTCA4CATA::createFactorMap(dataMeans,
```

```
                                        # use the constraint from the main map
                                        constraints = resMFA$Plotting.Data$constraints,
                                        col.points = col4Means,
                                        cex = 5,  # size of the dot (bigger)
                                        col.labels = col4Means,axis1 = 1, axis2 = 2,
                                        text.cex = 4)
MapGroup22 <- PTCA4CATA::createFactorMap(dataMeans,
                                        # use the constraint from the main map
                                        constraints = resMFA$Plotting.Data$constraints,
                                        col.points = col4Means,
                                        cex = 5,  # size of the dot (bigger)
                                        col.labels = col4Means,axis1 = 2, axis2 = 3,
                                        text.cex = 4)

# The map with observations and group means
a003.Map.I.withMeans <-  baseMap.j$zeMap_background +
  MapGroup$zeMap_dots + MapGroup$zeMap_text + label4Map
print(a003.Map.I.withMeans)
```




```
#_____________________________________________________________________
# Create the ellipses
# Bootstrapped CI ----
#_____________________________________________________________________
# Create Confidence Interval Plots
# use function MakeCIEllipses from package PTCA4CATA
GraphElli <- PTCA4CATA::MakeCIEllipses(BootCube.Gr$BootCube[,1:2,],
```

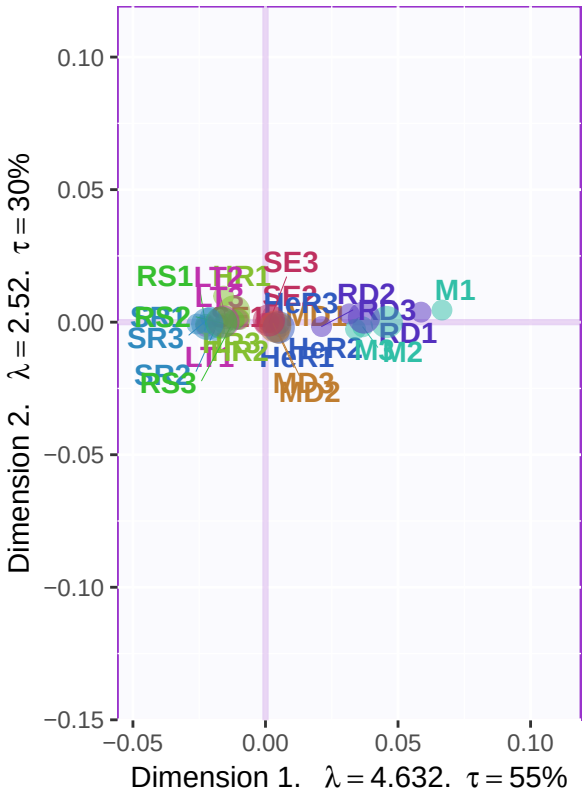
```
                                        names.of.factors = c("Dimension 1","Dimension 2"),
                                        col = col4Means,
                                        p.level = .95,axis1 = 1, axis2 = 2
)
GraphElli22 <- PTCA4CATA::MakeCIEllipses(BootCube.Gr$BootCube[,2:3,],
                                        names.of.factors = c("Dimension 2","Dimension 3"),
                                        col = col4Means,
                                        p.level = .95
)
#-----------------------------------------------------------------------
# create the I-map with Observations, means and confidence intervals
#
a004.Map.I.withCI <-  baseMap.j$zeMap_background + MapGroup$zeMap_text +  GraphElli + MapGroup$zeMap_do
print(a004.Map.I.withCI)
```



Partial Factor Scores: Job Role

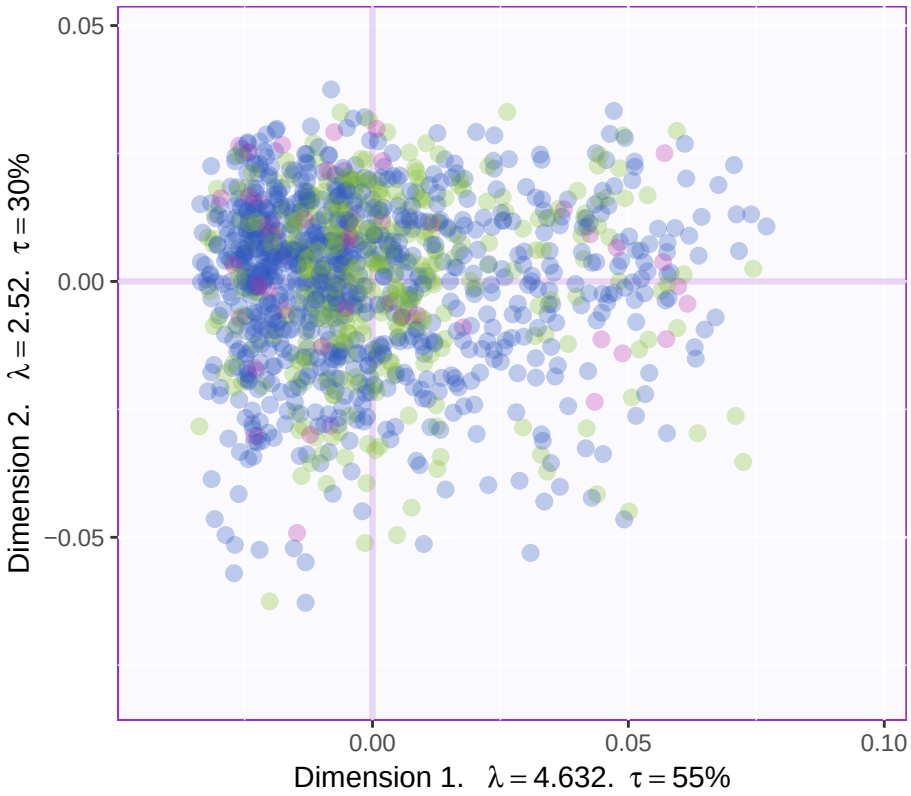
```
baseMap.j11 <- PTCA4CATA::createFactorMap(resMFA$mexPosition.Data$Table$partial.fi[1:1233,],
                                        col.labels = resMFA$Plotting.Data$fi.col,
                                        alpha.points =  .1,display.labels = FALSE)

dataMeans1 <- PTCA4CATA::getMeans(resMFA$mexPosition.Data$Table$partial.fi[1:1233,], data$JobRole)
dataMeans2 <- PTCA4CATA::getMeans(resMFA$mexPosition.Data$Table$partial.fi[1234:2466,], data$JobRole)
dataMeans3 <- PTCA4CATA::getMeans(resMFA$mexPosition.Data$Table$partial.fi[2467:3699,], data$JobRole)
#dataMeans4 <- dataMeans1 + dataMeans2 + dataMeans3
rownames(dataMeans1) <- c("HeR1","HR1","LT1","M1","MD1","RD1","RS1","SE1","SR1")
```

```r
rownames(dataMeans2) <- c("HeR2","HR2","LT2","M2","MD2","RD2","RS2","SE2","SR2")
rownames(dataMeans3) <- c("HeR3","HR3","LT3","M3","MD3","RD3","RS3","SE3","SR3")
col4data1 <- resMFA$Plotting.Data$fi.col
col4Means1 <- unique(col4data1)
# the map
MapGroup1 <- PTCA4CATA::createFactorMap(dataMeans1,
                                        # use the constraint from the main map
                                        constraints = resMFA$Plotting.Data$constraints,
                                        col.points = col4Means1,
                                        cex = 3,  # size of the dot (bigger)
                                        col.labels = col4Means1,display.labels = TRUE,
                                        text.cex = 4)
MapGroup2 <- PTCA4CATA::createFactorMap(dataMeans2,
                                        # use the constraint from the main map
                                        constraints = resMFA$Plotting.Data$constraints,
                                        col.points = col4Means1,
                                        cex = 3,  # size of the dot (bigger)
                                        col.labels = col4Means1, display.labels = TRUE,
                                        text.cex = 4)
MapGroup3 <- PTCA4CATA::createFactorMap(dataMeans3,
                                        # use the constraint from the main map
                                        constraints = resMFA$Plotting.Data$constraints,
                                        col.points = col4Means1,
                                        cex = 3,  # size of the dot (bigger)
                                        col.labels = col4Means1,display.labels = TRUE,
                                        text.cex = 4)

label4Map1 <- createxyLabels.gen(1,2,
                                 lambda = resMFA$mexPosition.Data$InnerProduct$eigs,
                                 tau = resMFA$mexPosition.Data$InnerProduct$t )

# The map with observations and group means
a003.Map.I.withMeans1 <- baseMap.j11$zeMap_background + label4Map1 + MapGroup1$zeMap_dots + MapGroup1$z
print(a003.Map.I.withMeans1)
```



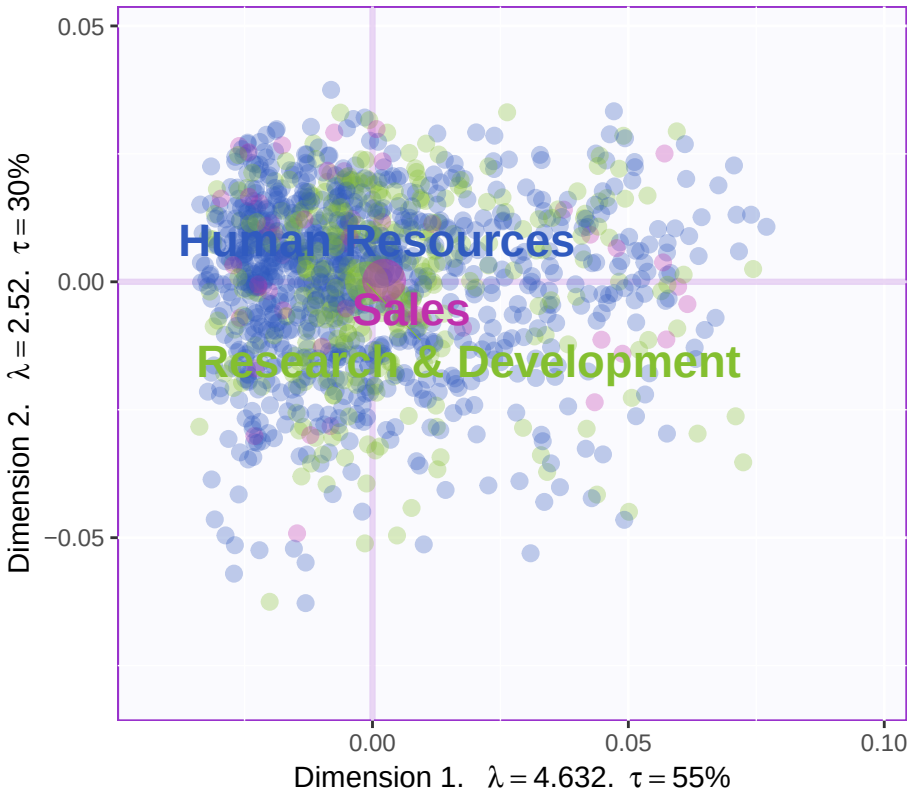
** Department **

```
baseMap.j0 <- PTCA4CATA::createFactorMap(resMFA2$mexPosition.Data$Table$fi,col.points = resMFA2$Plottin
                                         col.labels = resMFA2$Plotting.Data$fi.col,
                                         alpha.points =  .3,display.labels = FALSE)

label4Map0 <- createxyLabels.gen(1,2,
                                 lambda = resMFA2$mexPosition.Data$InnerProduct$eigs,
                                 tau = resMFA2$mexPosition.Data$InnerProduct$t )
# A graph for the J-set
b000.aggMap.j0 <- baseMap.j0$zeMap_background + # background layer
  baseMap.j0$zeMap_dots + baseMap.j0$zeMap_text + label4Map0
# We print this Map with the following code
print(b000.aggMap.j0)
```



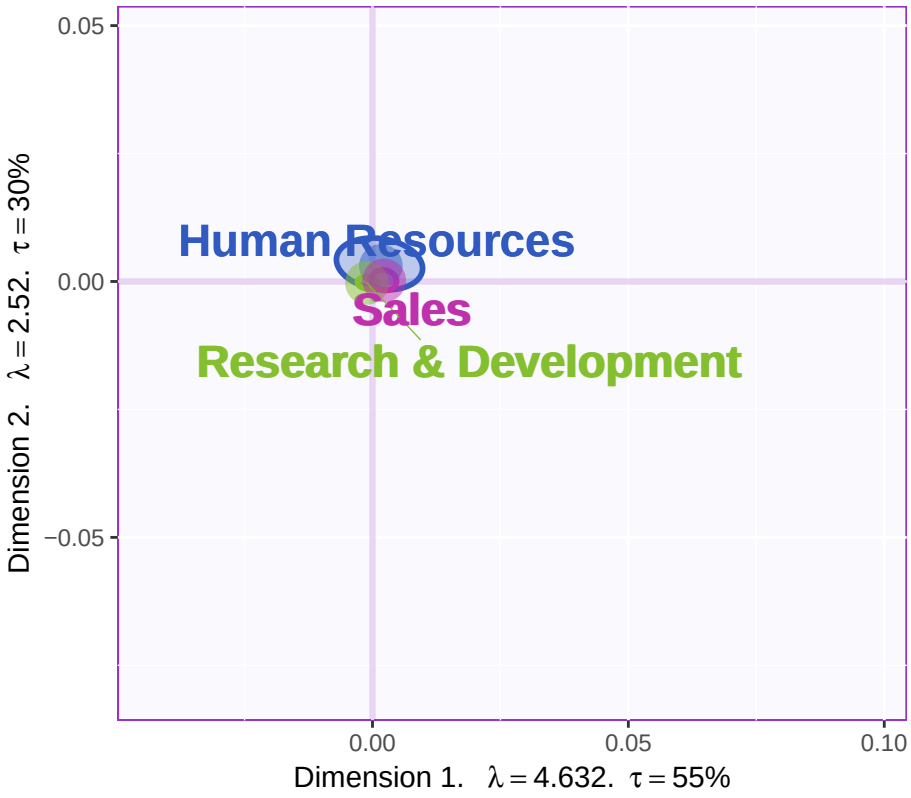

```
# Bootstrap for CI:
BootCube.Gr0 <- PTCA4CATA::Boot4Mean(resMFA2$mexPosition.Data$Table$fi,
                                     design = data$Department,
                                     niter = 100,
                                     suppressProgressBar = TRUE)
#_____________________________________________________________________
# Bootstrap ratios ----
bootRatios.Gr0 <- boot.ratio.test(BootCube.Gr0$BootCube)
#**********************************************************************
# eigenvalues: MonteCarlo Approach ----
#
random.eigen0 <- data4PCCAR::monteCarlo.eigen(X = data1, nIter = 100)
#
# eigenvalues: Bootstrap approach
#
bootstrap.eigen0 <- data4PCCAR::boot.eigen(data1, nIter = 100)
# Mean Map
#  create the map for the means
#  get the means by groups

dataMeans0 <- PTCA4CATA::getMeans(resMFA2$mexPosition.Data$Table$fi, data$Department)
# a vector of color for the means
col4data0 <- resMFA2$Plotting.Data$fi.col
col4Means0 <- unique(col4data0)
# the map
MapGroup0 <- PTCA4CATA::createFactorMap(dataMeans0,
```

```
                                    # use the constraint from the main map
                                    constraints = resMFA2$Plotting.Data$constraints,
                                    col.points = col4Means0,
                                    cex = 7,  # size of the dot (bigger)
                                    col.labels = col4Means0,
                                    text.cex = 6)
# The map with observations and group means
a003.Map.I.withMeans0 <- b000.aggMap.j0 +
  MapGroup0$zeMap_dots + MapGroup0$zeMap_text
print(a003.Map.I.withMeans0)
```




```
#_____________________________________________________________________
# Create the ellipses
# Bootstrapped CI ----
#_____________________________________________________________________
# Create Confidence Interval Plots
# use function MakeCIEllipses from package PTCA4CATA
GraphElli0 <- PTCA4CATA::MakeCIEllipses(BootCube.Gr0$BootCube[,1:2,],
                                    names.of.factors = c("Dimension 1","Dimension 2"),
                                    col = col4Means0,
                                    p.level = .95
)
#_____________________________________________________________________
# create the I-map with Observations, means and confidence intervals
#
a004.Map.I.withCI0 <-  baseMap.j0$zeMap_background + MapGroup0$zeMap_text +  GraphElli0 + MapGroup0$zeMa
```

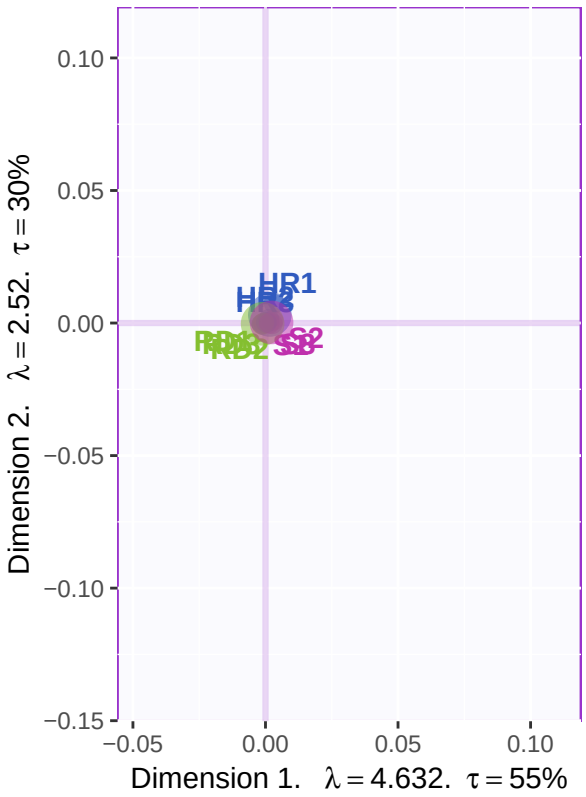
```
#_____________________________________________________________________
# plot it!
print(a004.Map.I.withCI0)
```



Partial Factor Scores: Department

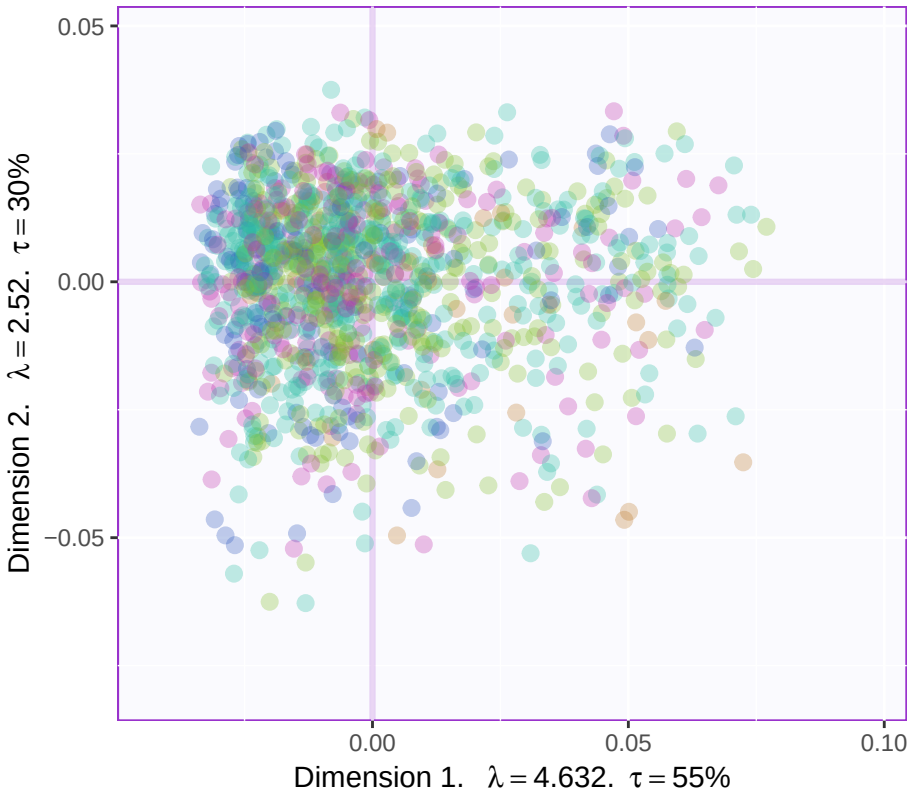
```
baseMap.j114 <- PTCA4CATA::createFactorMap(resMFA2$mexPosition.Data$Table$partial.fi[1:1233,],
                                           col.labels = resMFA2$Plotting.Data$fi.col,
                                           alpha.points =  .1,display.labels = FALSE)

dataMeans14 <- PTCA4CATA::getMeans(resMFA2$mexPosition.Data$Table$partial.fi[1:1233,], data$Department)
dataMeans24 <- PTCA4CATA::getMeans(resMFA2$mexPosition.Data$Table$partial.fi[1234:2466,], data$Departmer
dataMeans34 <- PTCA4CATA::getMeans(resMFA2$mexPosition.Data$Table$partial.fi[2467:3699,], data$Departmer
#dataMeans4 <- dataMeans1 + dataMeans2 + dataMeans3
rownames(dataMeans14) <- c("HR1","RD1","S1")
rownames(dataMeans24) <- c("HR2","RD2","S2")
rownames(dataMeans34) <- c("HR3","RD3","S3")

col4data4 <- resMFA2$Plotting.Data$fi.col
col4Means4 <- unique(col4data4)
# the map
MapGroup14 <- PTCA4CATA::createFactorMap(dataMeans14,
                                         # use the constraint from the main map
                                         constraints = resMFA2$Plotting.Data$constraints,
                                         col.points = col4Means4,
                                         cex = 3,  # size of the dot (bigger)
```

```
                                        col.labels = col4Means4,display.labels = TRUE,
                                        text.cex = 4)
MapGroup24 <- PTCA4CATA::createFactorMap(dataMeans24,
                                        # use the constraint from the main map
                                        constraints = resMFA2$Plotting.Data$constraints,
                                        col.points = col4Means4,
                                        cex = 3,  # size of the dot (bigger)
                                        col.labels = col4Means4, display.labels = TRUE,
                                        text.cex = 4)
MapGroup34 <- PTCA4CATA::createFactorMap(dataMeans34,
                                        # use the constraint from the main map
                                        constraints = resMFA2$Plotting.Data$constraints,
                                        col.points = col4Means4,
                                        cex = 3,  # size of the dot (bigger)
                                        col.labels = col4Means4,display.labels = TRUE,
                                        text.cex = 4)

label4Map14 <- createxyLabels.gen(1,2,
                                  lambda = resMFA2$mexPosition.Data$InnerProduct$eigs,
                                  tau = resMFA2$mexPosition.Data$InnerProduct$t )

# The map with observations and group means
a003.Map.I.withMeans14 <- baseMap.j114$zeMap_background + label4Map14 + MapGroup14$zeMap_dots + MapGrou
print(a003.Map.I.withMeans14)
```



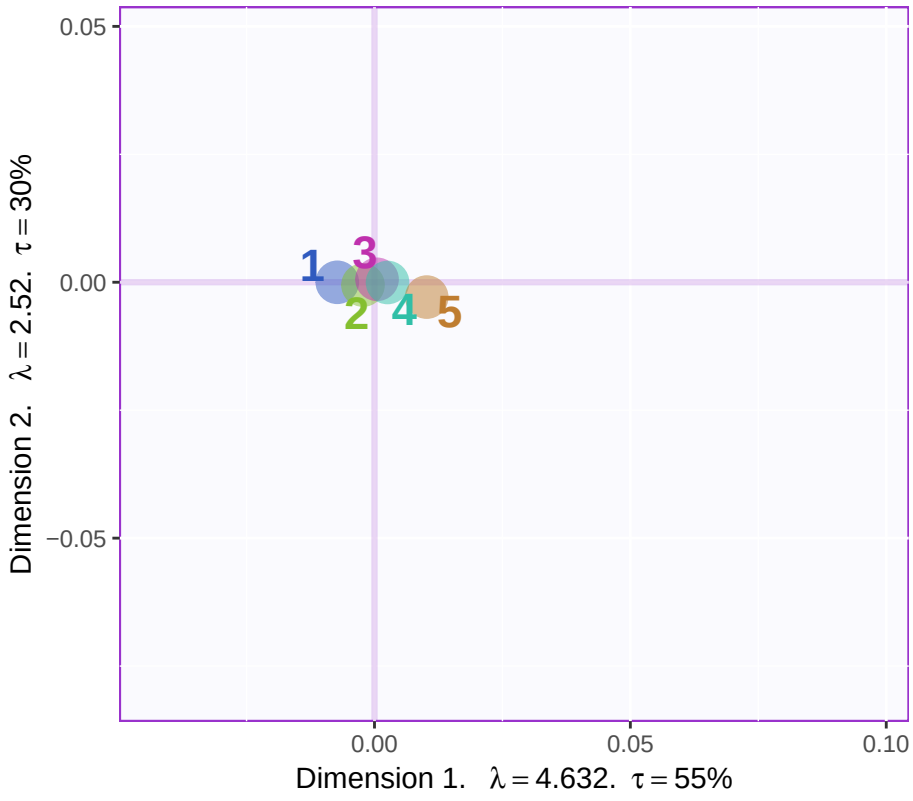
Education

```r
baseMap.j0 <- PTCA4CATA::createFactorMap(resMFA3$mexPosition.Data$Table$fi,col.points = resMFA3$Plotting
                                         col.labels = resMFA3$Plotting.Data$fi.col,
                                         alpha.points =  .3,display.labels = FALSE)

label4Map0 <- createxyLabels.gen(1,2,
                                 lambda = resMFA3$mexPosition.Data$InnerProduct$eigs,
                                 tau = resMFA3$mexPosition.Data$InnerProduct$t )
# A graph for the J-set
b000.aggMap.j0 <- baseMap.j0$zeMap_background + # background layer
  baseMap.j0$zeMap_dots + baseMap.j0$zeMap_text + label4Map0
# We print this Map with the following code
print(b000.aggMap.j0)
```



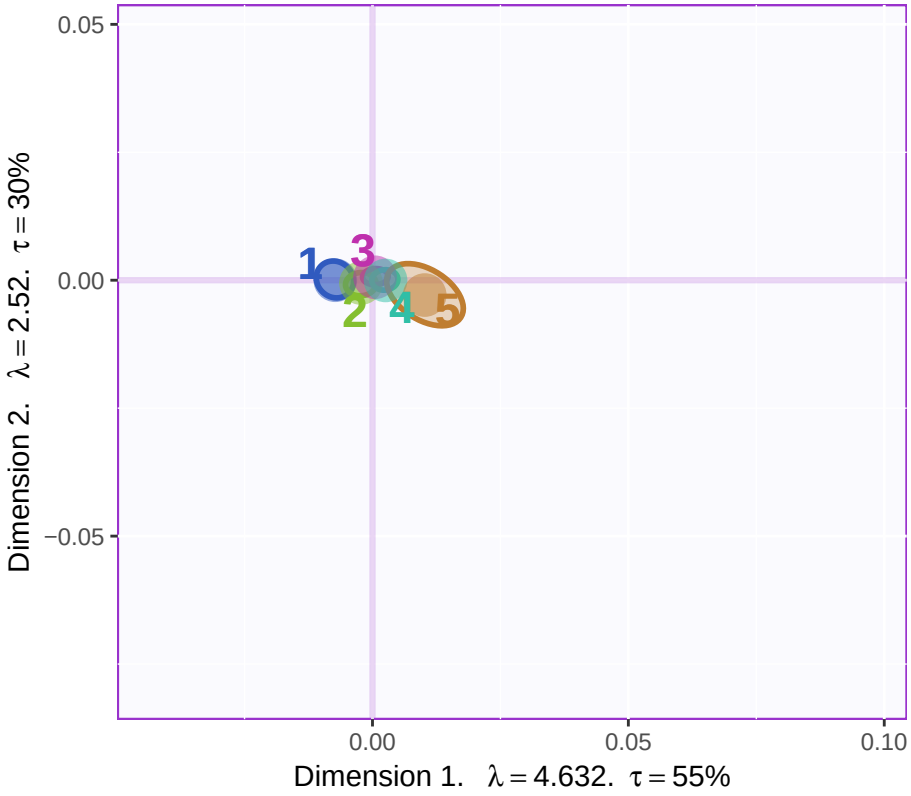

```r
# Bootstrap for CI:
BootCube.Gr0 <- PTCA4CATA::Boot4Mean(resMFA3$mexPosition.Data$Table$fi,
                                     design = data$Education,
                                     niter = 100,
                                     suppressProgressBar = TRUE)
#_____________________________________________________________________
# Bootstrap ratios ----
bootRatios.Gr0 <- boot.ratio.test(BootCube.Gr0$BootCube)
#**********************************************************************
# eigenvalues: MonteCarlo Approach ----
#
random.eigen0 <- data4PCCAR::monteCarlo.eigen(X = data1, nIter = 100)
#
```

```r
# eigenvalues: Bootstrap approach
#
bootstrap.eigen0 <- data4PCCAR::boot.eigen(data1, nIter = 100)
# Mean Map
#  create the map for the means
#  get the means by groups

dataMeans0 <- PTCA4CATA::getMeans(resMFA3$mexPosition.Data$Table$fi, data$Education)
# a vector of color for the means
col4data0 <- resMFA3$Plotting.Data$fi.col
col4Means0 <- unique(col4data0)
# the map
MapGroup0 <- PTCA4CATA::createFactorMap(dataMeans0,
                                        # use the constraint from the main map
                                        constraints = resMFA3$Plotting.Data$constraints,
                                        col.points = col4Means0,
                                        cex = 7,  # size of the dot (bigger)
                                        col.labels = col4Means0,
                                        text.cex = 6)
# The map with observations and group means
a003.Map.I.withMeans0 <- baseMap.j0$zeMap_background  +
  MapGroup0$zeMap_dots + MapGroup0$zeMap_text + label4Map0
print(a003.Map.I.withMeans0)
```




```r
#_____________________________________________________________
# Create the ellipses
```

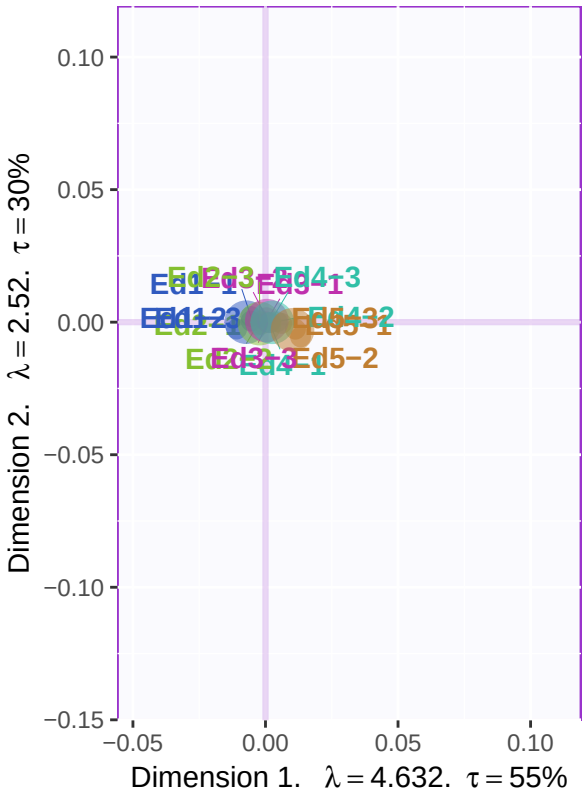
```
# Bootstrapped CI ----
#_____________________________________________________________________
# Create Confidence Interval Plots
# use function MakeCIEllipses from package PTCA4CATA
GraphElli0 <- PTCA4CATA::MakeCIEllipses(BootCube.Gr0$BootCube[,1:2,],
                                        names.of.factors = c("Dimension 1","Dimension 2"),
                                        col = col4Means0,
                                        p.level = .95
)
#_____________________________________________________________________
# create the I-map with Observations, means and confidence intervals
#
a004.Map.I.withCI0 <-  baseMap.j0$zeMap_background + MapGroup0$zeMap_text +  GraphElli0 +MapGroup0$zeMa
#_____________________________________________________________________
# plot it!
print(a004.Map.I.withCI0)
```



Partial Factor Scores: Education

```
baseMap.j12 <- PTCA4CATA::createFactorMap(resMFA3$mexPosition.Data$Table$partial.fi[1:1233,],
                                          col.labels = resMFA3$Plotting.Data$fi.col,
                                          alpha.points =  .1,display.labels = FALSE)

dataMeans12 <- PTCA4CATA::getMeans(resMFA3$mexPosition.Data$Table$partial.fi[1:1233,], data$Education)
dataMeans22 <- PTCA4CATA::getMeans(resMFA3$mexPosition.Data$Table$partial.fi[1234:2466,], data$Educatio
dataMeans32 <- PTCA4CATA::getMeans(resMFA3$mexPosition.Data$Table$partial.fi[2467:3699,], data$Educatio
```

```r
#dataMeans4 <- dataMeans1 + dataMeans2 + dataMeans3
rownames(dataMeans12) <- c("Ed1-1","Ed2-1","Ed3-1","Ed4-1","Ed5-1")
rownames(dataMeans22) <- c("Ed1-2","Ed2-2","Ed3-2","Ed4-2","Ed5-2")
rownames(dataMeans32) <- c("Ed1-3","Ed2-3","Ed3-3","Ed4-3","Ed5-3")

col4data2 <- resMFA3$Plotting.Data$fi.col
col4Means2 <- unique(col4data2)
# the map
MapGroup12 <- PTCA4CATA::createFactorMap(dataMeans12,
                                         # use the constraint from the main map
                                         constraints = resMFA3$Plotting.Data$constraints,
                                         col.points = col4Means2,
                                         cex = 3,  # size of the dot (bigger)
                                         col.labels = col4Means2,display.labels = TRUE,
                                         text.cex = 4)
MapGroup22 <- PTCA4CATA::createFactorMap(dataMeans22,
                                         # use the constraint from the main map
                                         constraints = resMFA3$Plotting.Data$constraints,
                                         col.points = col4Means2,
                                         cex = 3,  # size of the dot (bigger)
                                         col.labels = col4Means2, display.labels = TRUE,
                                         text.cex = 4)
MapGroup32 <- PTCA4CATA::createFactorMap(dataMeans32,
                                         # use the constraint from the main map
                                         constraints = resMFA3$Plotting.Data$constraints,
                                         col.points = col4Means2,
                                         cex = 3,  # size of the dot (bigger)
                                         col.labels = col4Means2,display.labels = TRUE,
                                         text.cex = 4)

label4Map12 <- createxyLabels.gen(1,2,
                                  lambda = resMFA3$mexPosition.Data$InnerProduct$eigs,
                                  tau = resMFA3$mexPosition.Data$InnerProduct$t )

# The map with observations and group means
a003.Map.I.withMeans12 <- baseMap.j12$zeMap_background + label4Map12 + MapGroup12$zeMap_dots + MapGroup
print(a003.Map.I.withMeans12)
```



Business Travel

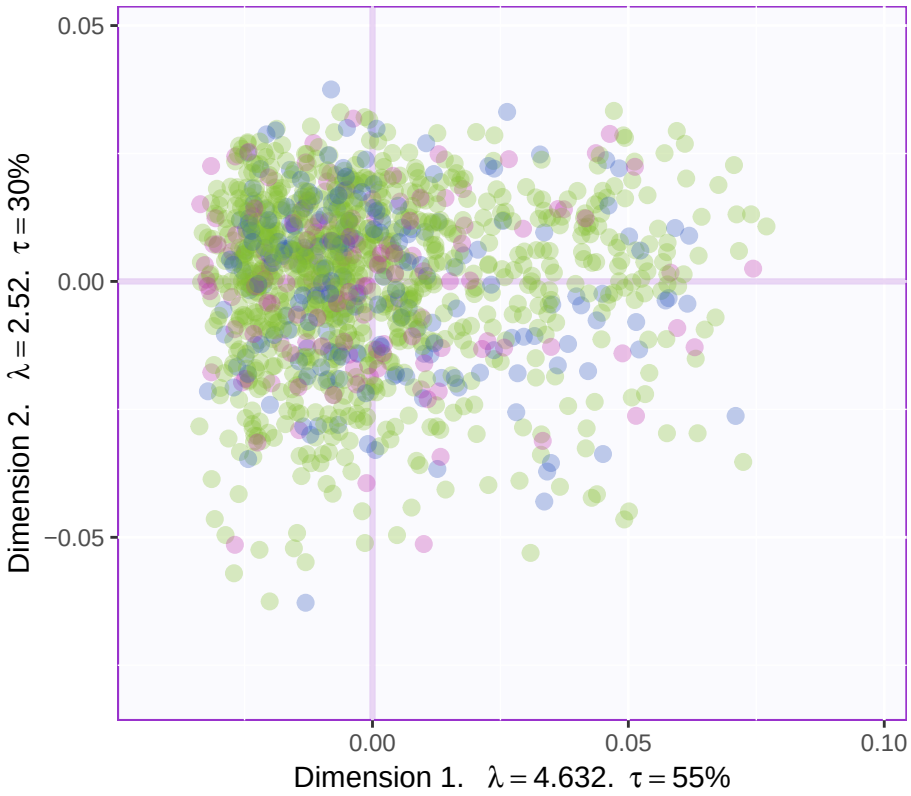
```
resMFA4 <- mpMFA(data1 ,column.design = a,graphs = FALSE,DESIGN = data$BusinessTravel)
```

```
## [1] "Preprocessed the Rows of the data matrix using:  None"
## [1] "Preprocessed the Columns of the data matrix using:  Center_1Norm"
## [1] "Preprocessed the Tables of the data matrix using:  MFA_Normalization"
## [1] "Preprocessing Completed"
## [1] "Optimizing using:  None"
```

```
## Warning in sqrt(eigOut$values): NaNs produced
```

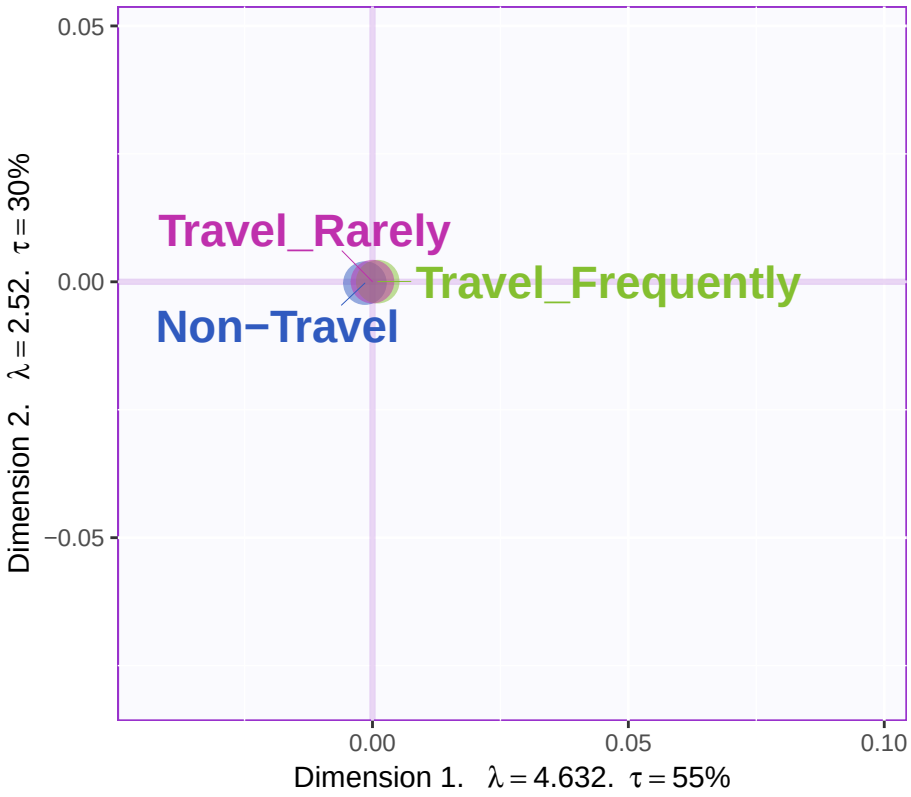
```
## [1] "Processing Complete"
```

```
baseMap.j0 <- PTCA4CATA::createFactorMap(resMFA4$mexPosition.Data$Table$fi,col.points = resMFA4$Plottin
                                         col.labels = resMFA4$Plotting.Data$fi.col,
                                         alpha.points =  .3,display.labels = FALSE)

label4Map0 <- createxyLabels.gen(1,2,
                                 lambda = resMFA4$mexPosition.Data$InnerProduct$eigs,
                                 tau = resMFA4$mexPosition.Data$InnerProduct$t )
# A graph for the J-set
b000.aggMap.j0 <- baseMap.j0$zeMap_background + # background layer
  baseMap.j0$zeMap_dots + baseMap.j0$zeMap_text + label4Map0
# We print this Map with the following code
print(b000.aggMap.j0)
```



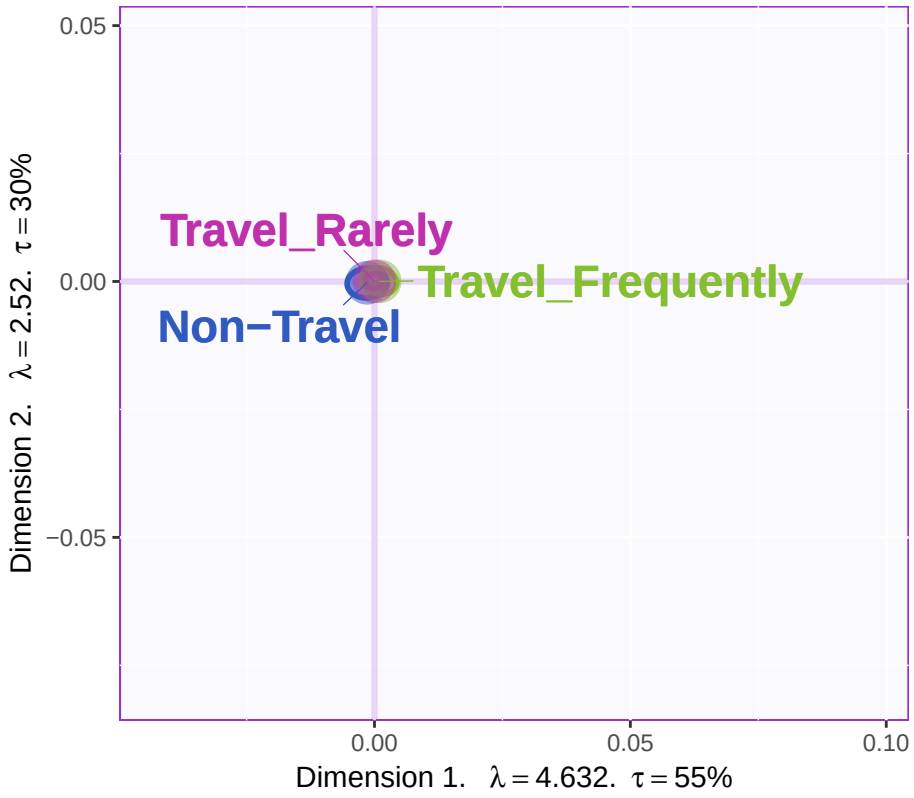

```
# Bootstrap for CI:
BootCube.Gr0 <- PTCA4CATA::Boot4Mean(resMFA4$mexPosition.Data$Table$fi,
                                     design = data$BusinessTravel,
                                     niter = 100,
                                     suppressProgressBar = TRUE)
#_____________________________________________________________________
# Bootstrap ratios ----
bootRatios.Gr0 <- boot.ratio.test(BootCube.Gr0$BootCube)
#**********************************************************************
# eigenvalues: MonteCarlo Approach ----
#
random.eigen0 <- data4PCCAR::monteCarlo.eigen(X = data1, nIter = 100)
#
# eigenvalues: Bootstrap approach
#
bootstrap.eigen0 <- data4PCCAR::boot.eigen(data1, nIter = 100)
# Mean Map
#  create the map for the means
#  get the means by groups

dataMeans0 <- PTCA4CATA::getMeans(resMFA4$mexPosition.Data$Table$fi, data$BusinessTravel)
# a vector of color for the means
col4data0 <- resMFA4$Plotting.Data$fi.col
col4Means0 <- unique(col4data0)
# the map
MapGroup0 <- PTCA4CATA::createFactorMap(dataMeans0,
```

```
                                    # use the constraint from the main map
                                    constraints = resMFA4$Plotting.Data$constraints,
                                    col.points = col4Means0,
                                    cex = 7,  # size of the dot (bigger)
                                    col.labels = col4Means0,
                                    text.cex = 6)
# The map with observations and group means
a003.Map.I.withMeans0 <- baseMap.j0$zeMap_background  +
  MapGroup0$zeMap_dots + MapGroup0$zeMap_text + label4Map0
print(a003.Map.I.withMeans0)
```




```
#_____________________________________________________________________
# Create the ellipses
# Bootstrapped CI ----
#_____________________________________________________________________
# Create Confidence Interval Plots
# use function MakeCIEllipses from package PTCA4CATA
GraphElli0 <- PTCA4CATA::MakeCIEllipses(BootCube.Gr0$BootCube[,1:2,],
                                    names.of.factors = c("Dimension 1","Dimension 2"),
                                    col = col4Means0,
                                    p.level = .95
)
#_____________________________________________________________________
# create the I-map with Observations, means and confidence intervals
#
a004.Map.I.withCI0 <-  baseMap.j0$zeMap_background + MapGroup0$zeMap_text +  GraphElli0 + MapGroup0$zeMa
```

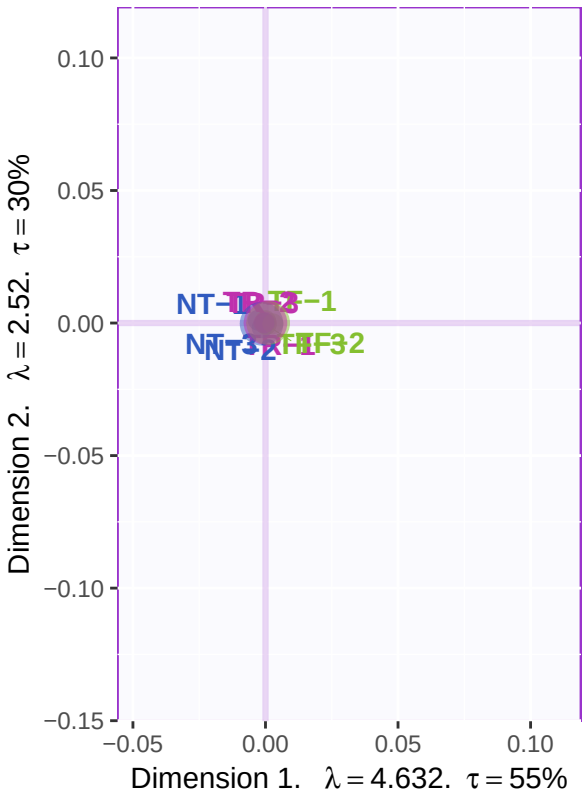
```
#_____________________________________________________________________
# plot it!
print(a004.Map.I.withCI0)
```



Partial Factor Scores: Business Travel

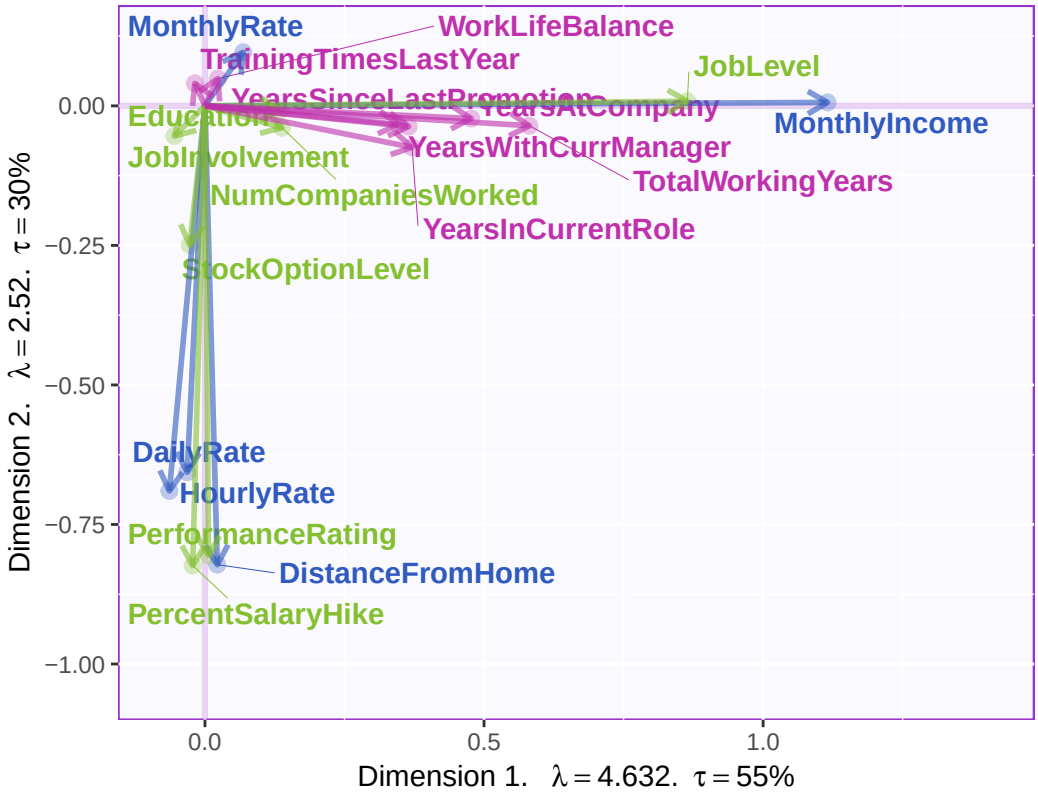
```
baseMap.j113 <- PTCA4CATA::createFactorMap(resMFA4$mexPosition.Data$Table$partial.fi[1:1233,],
                                           col.labels = resMFA4$Plotting.Data$fi.col,
                                           alpha.points =  .1,display.labels = FALSE)

dataMeans13 <- PTCA4CATA::getMeans(resMFA4$mexPosition.Data$Table$partial.fi[1:1233,], data$BusinessTra
dataMeans23 <- PTCA4CATA::getMeans(resMFA4$mexPosition.Data$Table$partial.fi[1234:2466,], data$Business
dataMeans33 <- PTCA4CATA::getMeans(resMFA4$mexPosition.Data$Table$partial.fi[2467:3699,], data$Business
#dataMeans4 <- dataMeans1 + dataMeans2 + dataMeans3
rownames(dataMeans13) <- c("NT-1","TF-1","TR-1")
rownames(dataMeans23) <- c("NT-2","TF-2","TR-2")
rownames(dataMeans33) <- c("NT-3","TF-3","TR-3")

col4data3 <- resMFA4$Plotting.Data$fi.col
col4Means3 <- unique(col4data3)
# the map
MapGroup13 <- PTCA4CATA::createFactorMap(dataMeans13,
                                         # use the constraint from the main map
                                         constraints = resMFA4$Plotting.Data$constraints,
                                         col.points = col4Means3,
                                         cex = 3,  # size of the dot (bigger)
```

```
                                        col.labels = col4Means3,display.labels = TRUE,
                                        text.cex = 4)
MapGroup23 <- PTCA4CATA::createFactorMap(dataMeans23,
                                        # use the constraint from the main map
                                        constraints = resMFA4$Plotting.Data$constraints,
                                        col.points = col4Means3,
                                        cex = 3,  # size of the dot (bigger)
                                        col.labels = col4Means3, display.labels = TRUE,
                                        text.cex = 4)
MapGroup33 <- PTCA4CATA::createFactorMap(dataMeans33,
                                        # use the constraint from the main map
                                        constraints = resMFA4$Plotting.Data$constraints,
                                        col.points = col4Means3,
                                        cex = 3,  # size of the dot (bigger)
                                        col.labels = col4Means3,display.labels = TRUE,
                                        text.cex = 4)

label4Map13 <- createxyLabels.gen(1,2,
                                  lambda = resMFA4$mexPosition.Data$InnerProduct$eigs,
                                  tau = resMFA4$mexPosition.Data$InnerProduct$t )

# The map with observations and group means
a003.Map.I.withMeans13 <- baseMap.j113$zeMap_background + label4Map13 + MapGroup13$zeMap_dots + MapGrou
print(a003.Map.I.withMeans13)
```



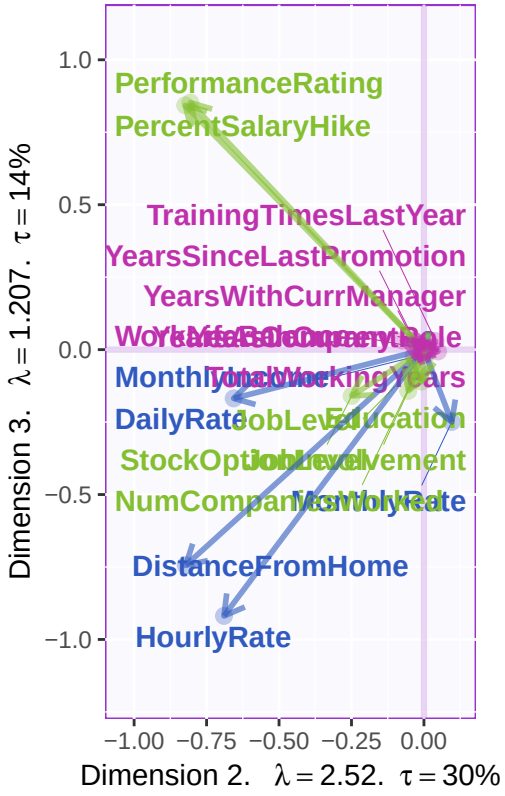
8.3 Loadings

```
baseMap.j2 <- PTCA4CATA::createFactorMap(resMFA$mexPosition.Data$Table$Q,col.points = resMFA$Plotting.Da
                                         col.labels =resMFA$Plotting.Data$fj.col ,
                                         alpha.points =  .3,display.labels = TRUE)
# arrows
zeArrows2 <- addArrows(resMFA$mexPosition.Data$Table$Q , col = resMFA$Plotting.Data$fj.col )
# A graph for the J-set
b000.aggMap.j2 <- baseMap.j2$zeMap_background + # background layer
  baseMap.j2$zeMap_dots + baseMap.j2$zeMap_text +  # dots & labels
  label4Map + zeArrows2
# We print this Map with the following code
print(b000.aggMap.j2)
```



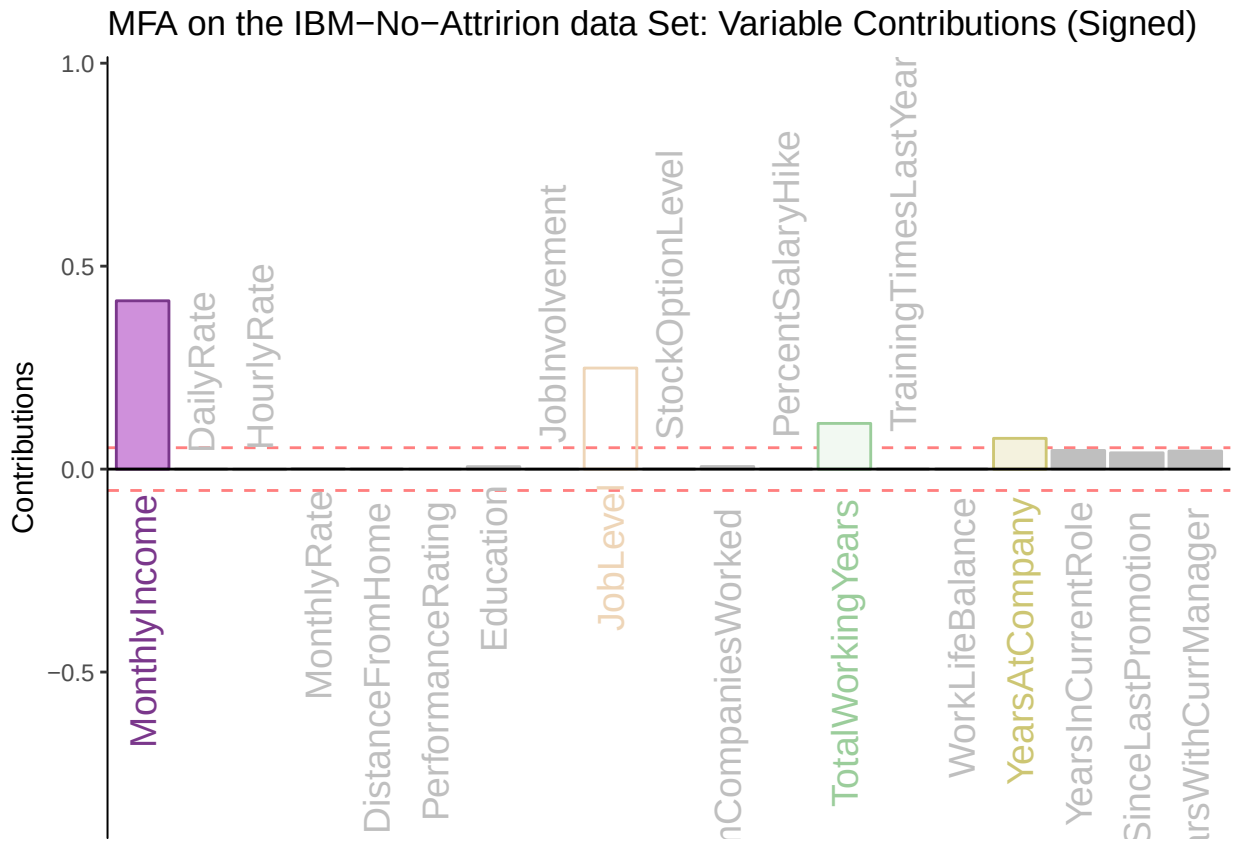

```
baseMap.j21 <- PTCA4CATA::createFactorMap(resMFA$mexPosition.Data$Table$Q,col.points = resMFA$Plotting.
                                         col.labels =resMFA$Plotting.Data$fj.col ,
                                         alpha.points =  .3,display.labels = TRUE,axis1 = 2,axis2 = 3)
# arrows
label4Map3 <- createxyLabels.gen(2,3,
                                 lambda = resMFA$mexPosition.Data$InnerProduct$eigs,
                                 tau = resMFA$mexPosition.Data$InnerProduct$t )

zeArrows3 <- addArrows(resMFA$mexPosition.Data$Table$Q , col = resMFA$Plotting.Data$fj.col,axis1 = 2,ax
# A graph for the J-set
b000.aggMap.j21 <- baseMap.j21$zeMap_background + # background layer
  baseMap.j21$zeMap_dots + baseMap.j21$zeMap_text +  # dots & labels
```

```
  label4Map3 + zeArrows3
# We print this Map with the following code
print(b000.aggMap.j21)
```

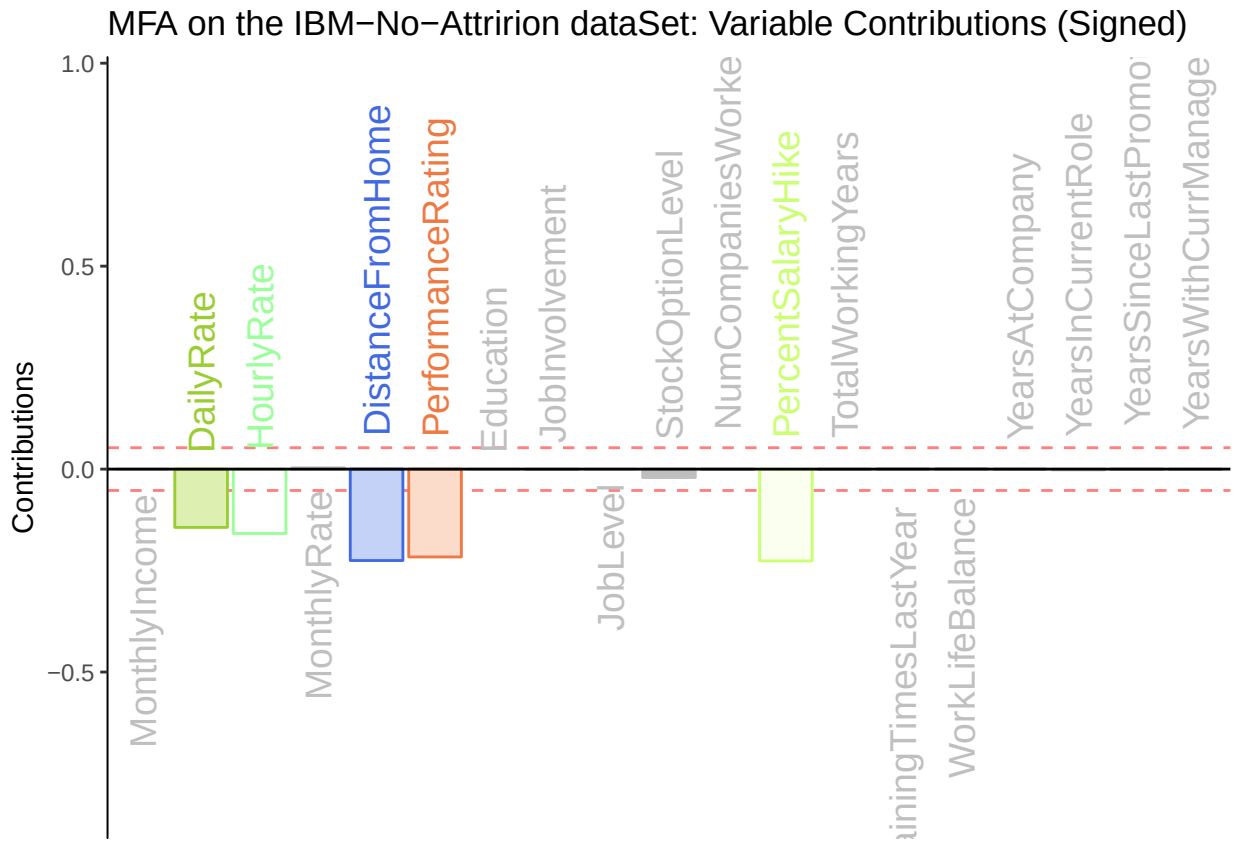


8.4 Contribution

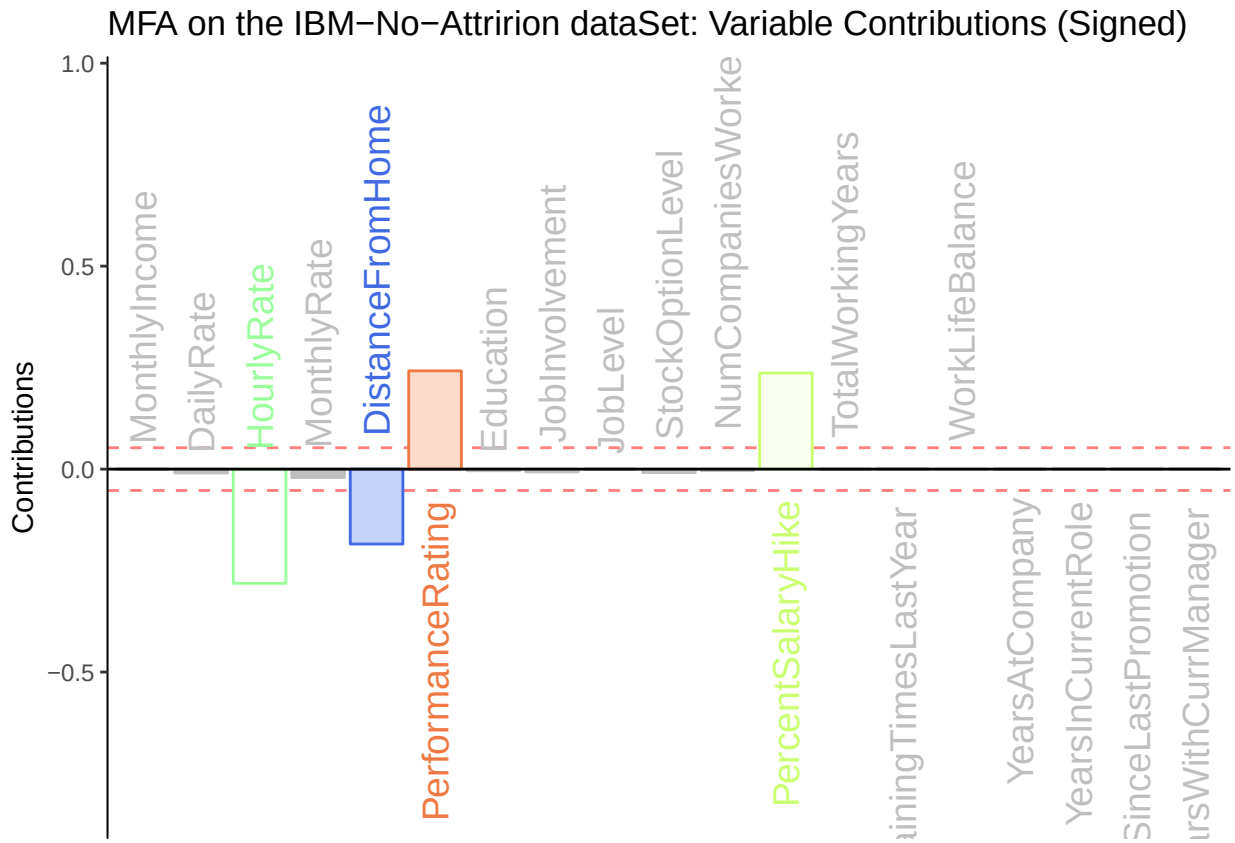
```
col4J.ibm <- prettyGraphsColorSelection(NCOL(data1))
signed.ctrJ <- resMFA$mexPosition.Data$Table$cj * sign(resMFA$mexPosition.Data$Table$Q)
b003.ctrJ.s.1 <- PrettyBarPlot2(signed.ctrJ[,1],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                                color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'MFA on the IBM-No-Attririon data Set: Variable Contributions (S
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ))
)
print(b003.ctrJ.s.1)
```




```
b004.ctrJ.s.2 <- PrettyBarPlot2(signed.ctrJ[,2],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                                color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'MFA on the IBM-No-Attririon dataSet: Variable Contributions (Si
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ))
)
print(b004.ctrJ.s.2)
```

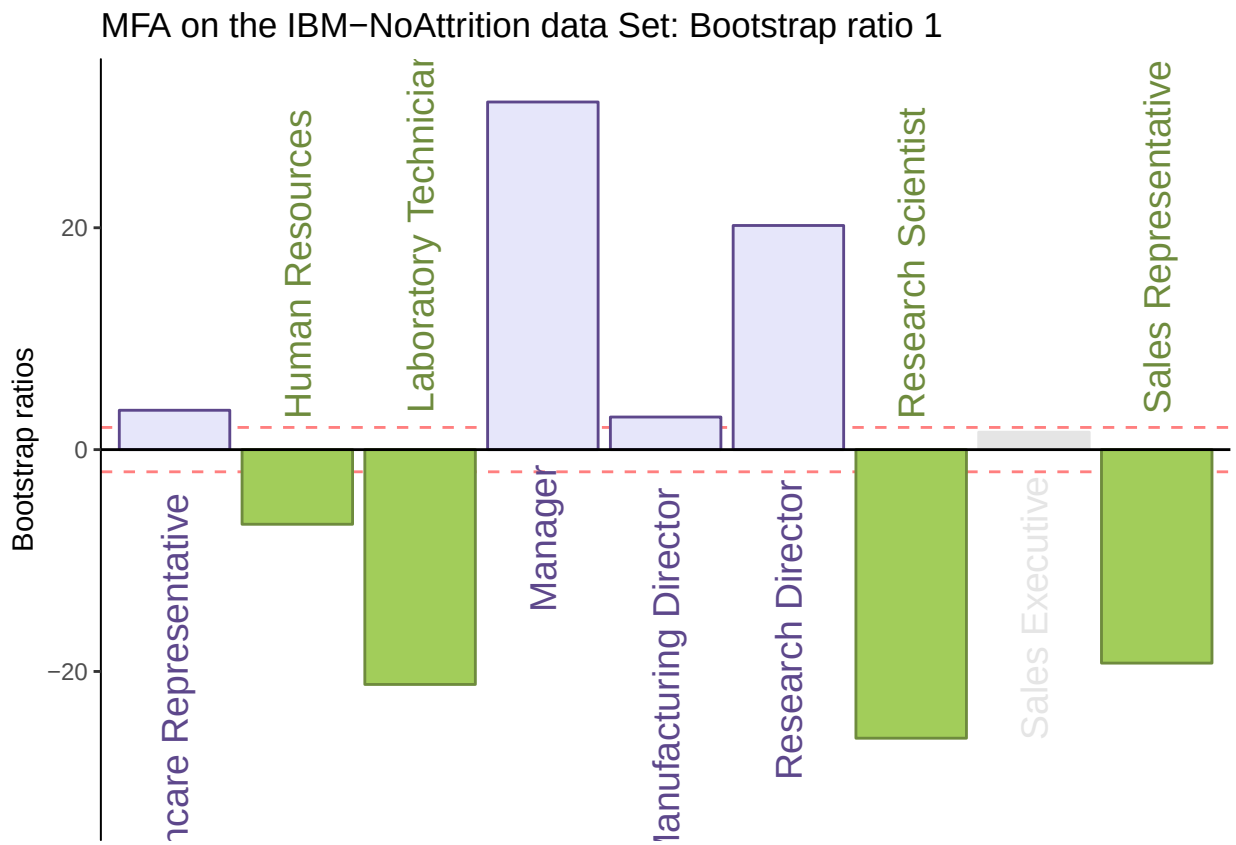



```
b004.ctrJ.s.3 <- PrettyBarPlot2(signed.ctrJ[,3],
                                threshold = 1 / NROW(signed.ctrJ),
                                font.size = 5,
                                color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                main = 'MFA on the IBM-No-Attririon dataSet: Variable Contributions (Si
                                ylab = 'Contributions',
                                ylim = c(1.2*min(signed.ctrJ), 1.2*max(signed.ctrJ))
)
print(b004.ctrJ.s.3)
```

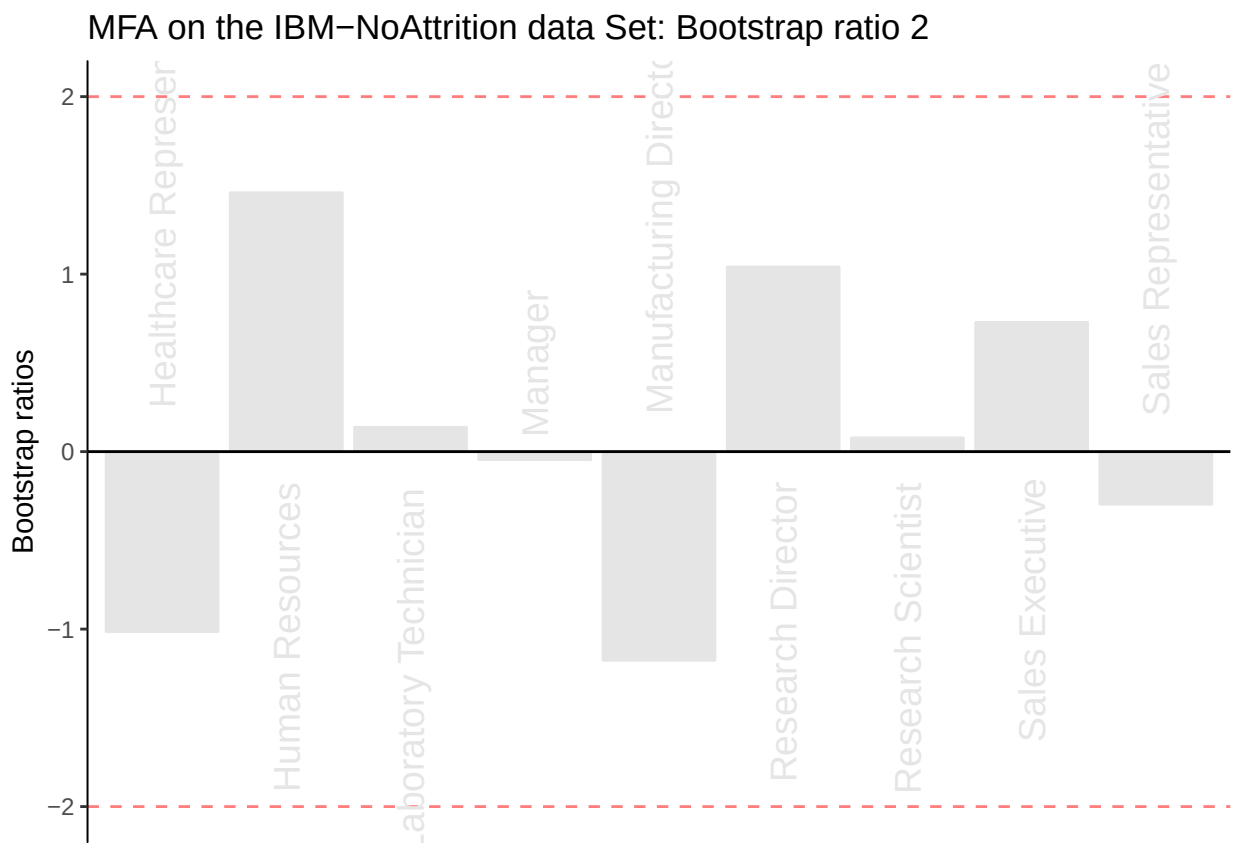


8.5 Bootstrap Ratios for Variables

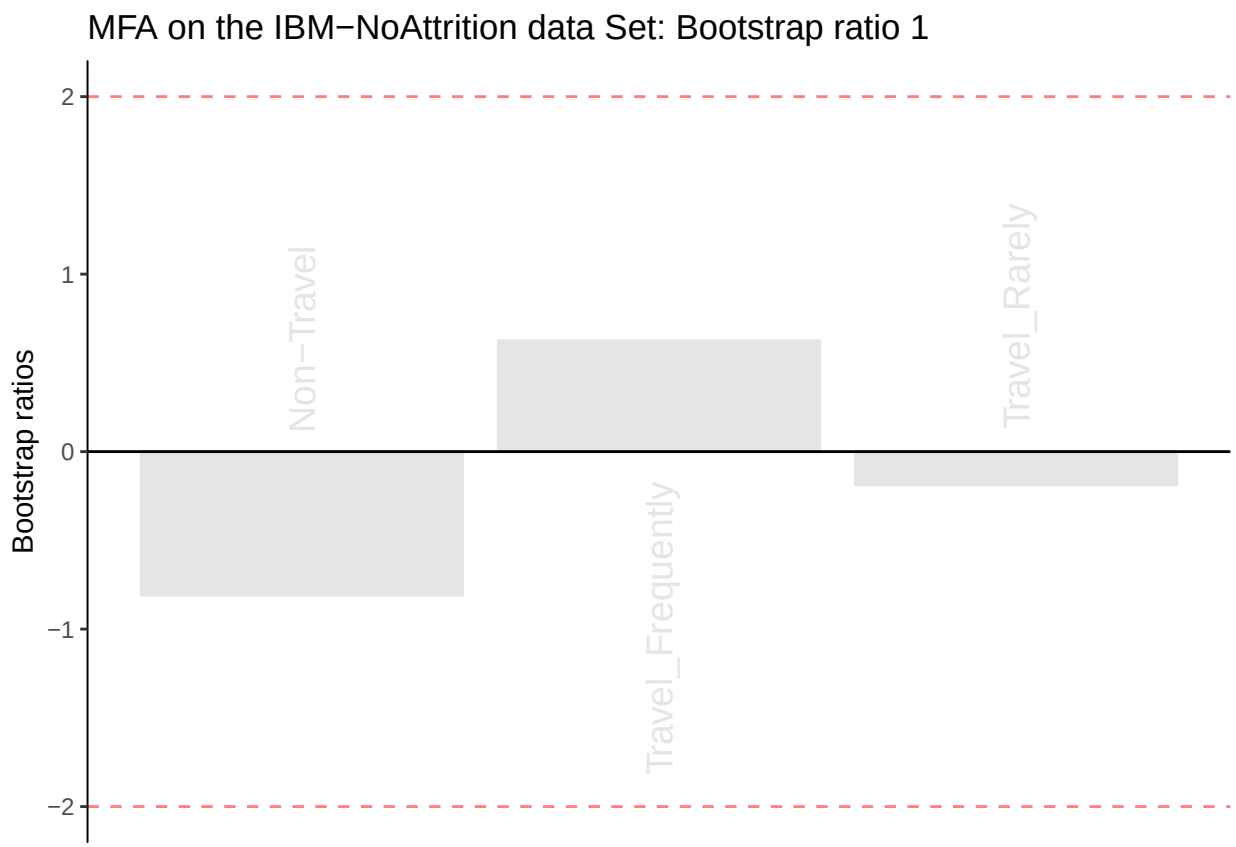
```
BR2 <- bootRatios.Gr$boot.ratios
ba001.BR111 <- PrettyBarPlot2(BR2[,1],
                              threshold = 2,
                              font.size = 5,
                              #color4bar = gplots::col2hex(col4J.ibm),
                              main = paste0( 'MFA on the IBM-NoAttrition data Set: Bootstrap ratio ',1),
                              ylab = 'Bootstrap ratios'
                              #ylim = c(1.2*min(BR[,laDim]), 1.2*max(BR[,laDim]))
)
print(ba001.BR111)
```



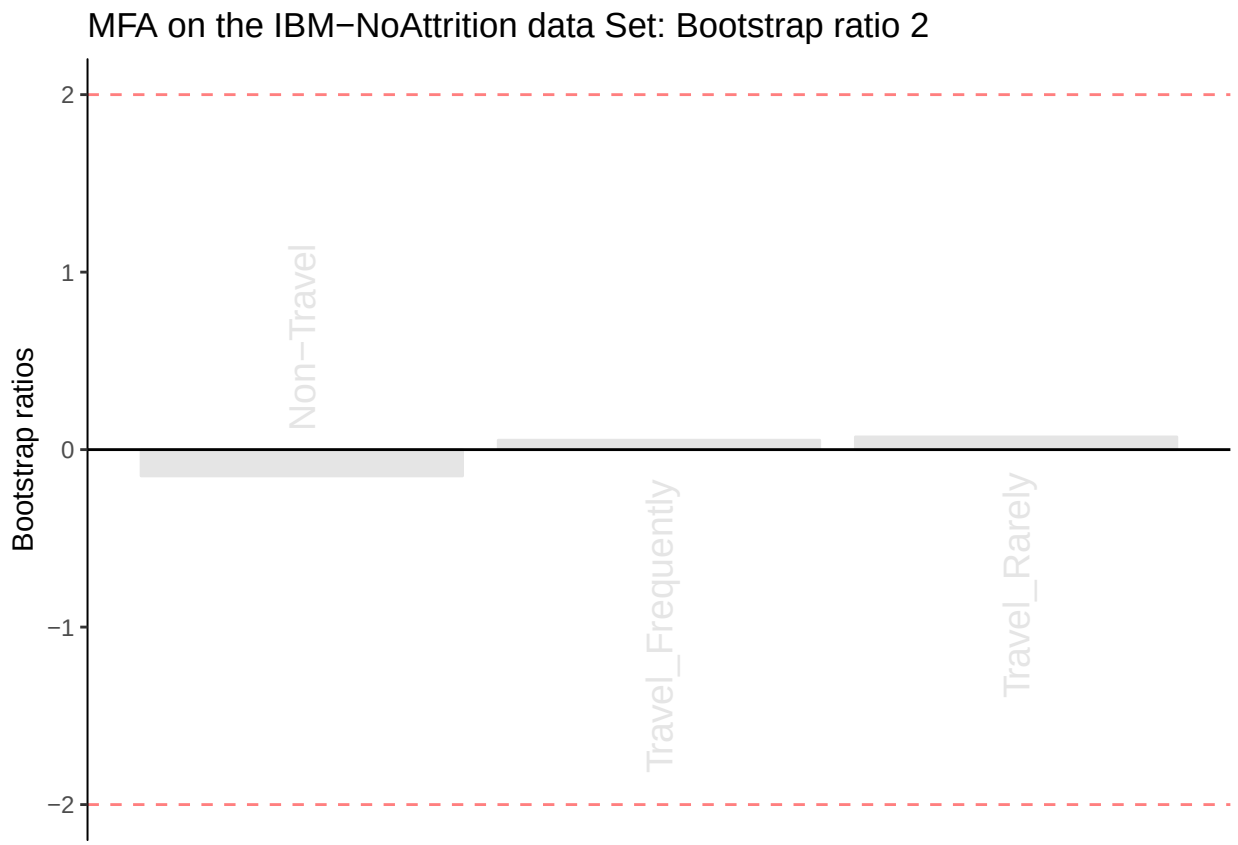

```
#
ba002.BR211 <- PrettyBarPlot2(BR2[,2],
                              threshold = 2,
                              font.size = 5,
                              #color4bar = gplots::col2hex(col4J.ibm),
                              main = paste0(
                                'MFA on the IBM-NoAttrition data Set: Bootstrap ratio ',2),
                              ylab = 'Bootstrap ratios'
)
print(ba002.BR211)
```




```
BR3 <- bootRatios.Gr0$boot.ratios
ba001.BR1113 <- PrettyBarPlot2(BR3[,1],
                               threshold = 2,
                               font.size = 5,
                               #color4bar = gplots::col2hex(col4J.ibm),
                               main = paste0( 'MFA on the IBM-NoAttrition data Set: Bootstrap ratio ',1),
                               ylab = 'Bootstrap ratios'
                               #ylim = c(1.2*min(BR[,laDim]), 1.2*max(BR[,laDim]))
)
print(ba001.BR1113)
```




```
#
ba002.BR2114 <- PrettyBarPlot2(BR3[,2],
                               threshold = 2,
                               font.size = 5,
                               #color4bar = gplots::col2hex(col4J.ibm),
                               main = paste0(
                                 'MFA on the IBM-NoAttrition data Set: Bootstrap ratio ',2),
                               ylab = 'Bootstrap ratios'
)
print(ba002.BR2114)
```



8.6 Summary

When we interpret the factor scores and loadings together, the MFA revealed almost the similar results as PCA:

- Usually the Research and Development people have more work-experience & Monthly Income in comparison to other department type
- Generally, Managers and Directors are the ones with PhD & Master's with higher job level, Sales representative and Lab Technician have no college education while Healthcare Representatives & Manufacturing Directors have Bachelor's degree
- People who are Managers and Research Directors have the highest pay and are more seasoned while people who are Sales representative and Lab Technician have low pay and less experience
- It is also observed that the Mangers and Research Directors travel the most, could be their meetings and conferences while the HR, Sales representative,Lab Technician, Research Scientists travel occasionally
- Also, from the Partial factor score plot we can see that the first table drives the mean Fi for both the Managers and the Research Directors also we can say that people from diverse backgrounds can be Managers & Research Directors
- For the factor plot the gender type does not show any significant inference as the mean are almost overlapping each other and for the department type similar results can be seen as R&D, Sales and HR all intersect their mean confidence Intervals.

```
options(knitr.duplicate.label = 'allow')
```

9 Correspondance Analysis

Correspondence Analysis is a generalized principal component analysis tailored for the analysis of qualitative data. Also commonly known as reciprocal averaging is a multivariate statistical technique proposed by Herman Otto. It is conceptually similar to principal component analysis, but applies to categorical rather than continuous data. In a similar manner to principal component analysis, it provides a means of displaying or summarising a set of data in two-dimensional graphical form.This is a descriptive/exploratory technique designed to analyze simple two-way and multi-way tables containing some measure of correspondence between the rows and columns. The results provide information which is similar in nature to those produced by Factor Analysis techniques, and they allow you to explore the structure of categorical variables included in the table.Originally, ca was created to analyze contingency tables, but, ca is so versatile that it is used with a lot of other data table types.

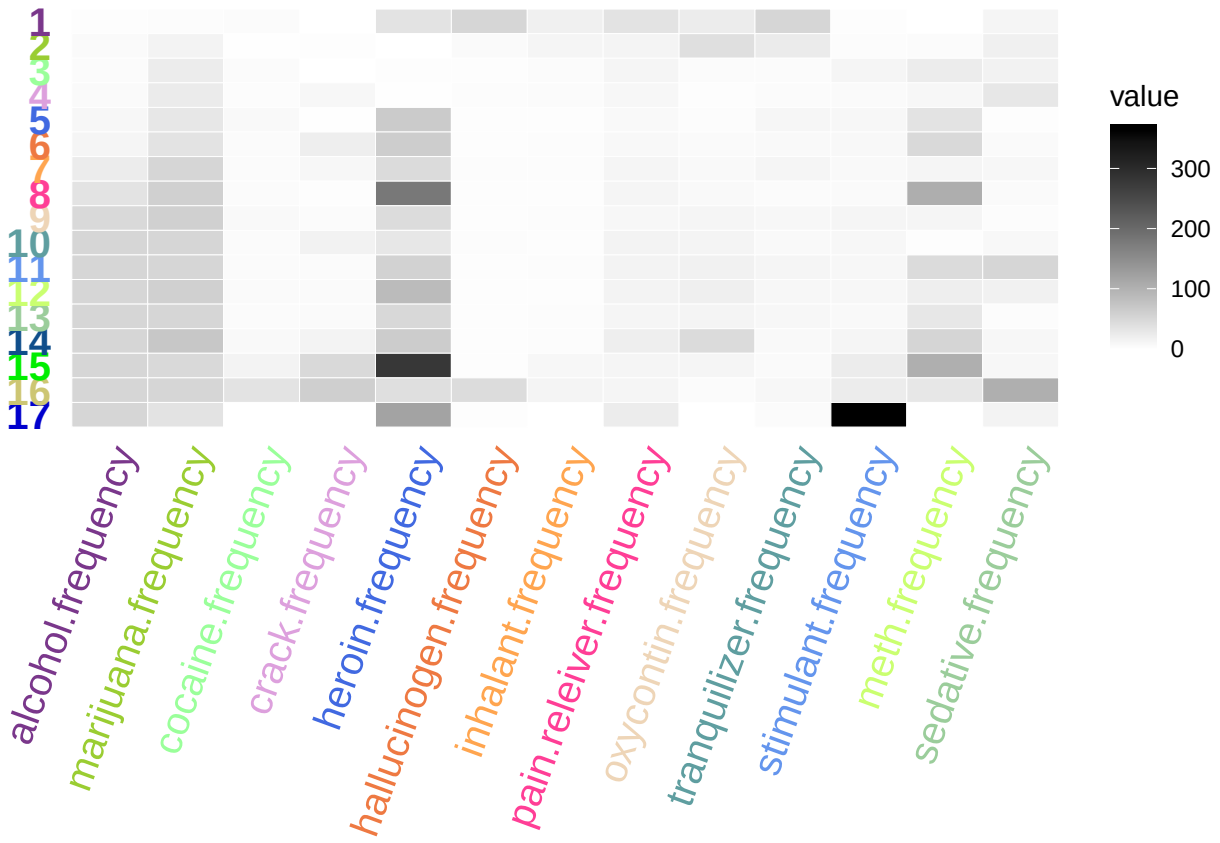
Dataset

The dataset is about Drug-use-by age which consists of 17 observations and 28 variables.This dataset describes about the variation of drugs people consume across all age groups. The dataset consists of both the frequency and the percentage of the drug. As,in statistics, a contingency table (also known as a cross tabulation or crosstab) is a type of table in a matrix format that displays the (multivariate) frequency distribution of the variables. So, I dropped the variables showing the percentage count of drug use and kept only the variables displaying the median of the frequency count of drug use across all age group.

Heat Map

A heat map (or heatmap) is a graphical representation of data where the individual values contained in a matrix are represented as colors.

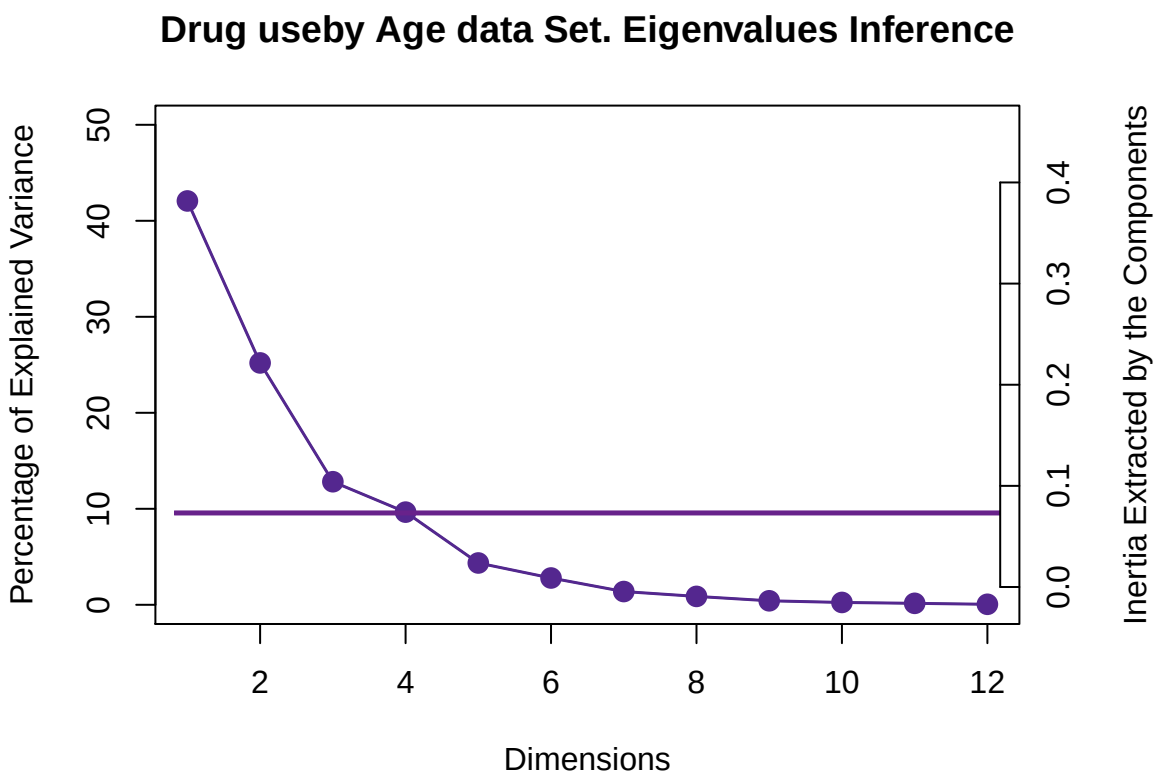
```
col4J.drugs <- prettyGraphsColorSelection(NCOL(x))
heatMapIJ <- makeggHeatMap4CT(x,
                 colorAttributes = col4J.drugs,
                 fontSize.x = 15)
print(heatMapIJ)
```



Scree plot

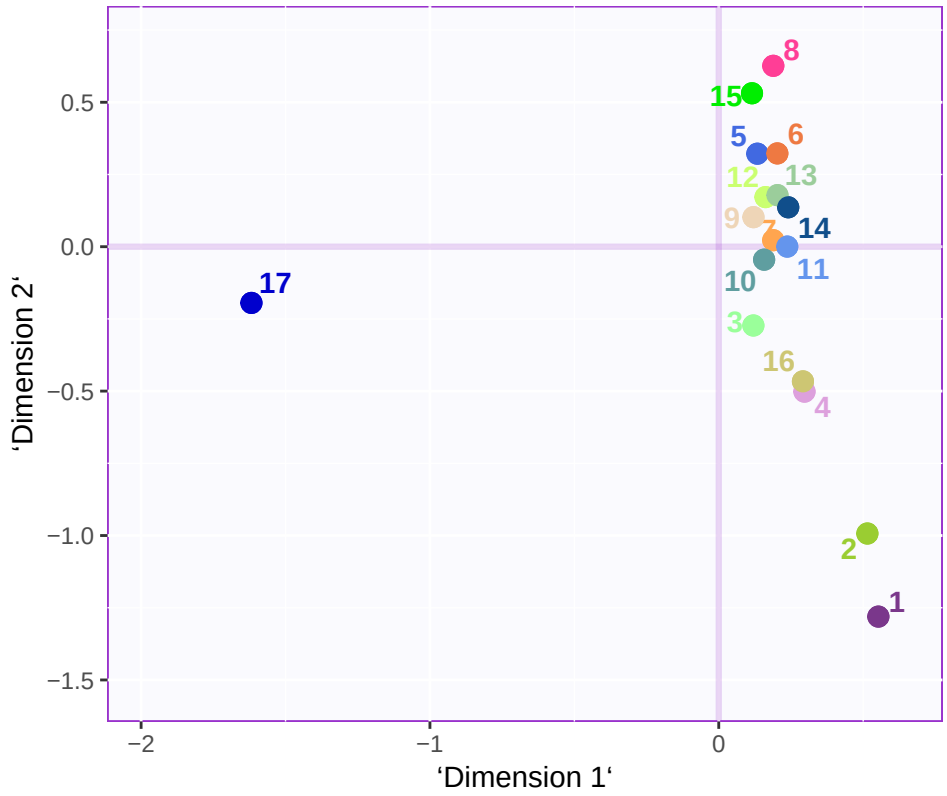
A Scree Plot is a simple line segment plot that shows the fraction of total variance in the data as explained or represented by each PC.`(In the PCA literature, the plot is called a 'Scree' Plot because it often looks like a 'scree' slope, where rocks have fallen down and accumulated on the side of a mountain.)`The scree plot shows the eigenvalues, the amount of information on each component. The number of components (the dimensionality of the factor space) is min(nrow(DATA), ncol(DATA)) minus 1.The scree plot is used to determine how many of the components should be interpreted. * `plot` draws the line that connects all data points by `type = "l"` * The first `points` function draws round purple dots. * The second `points` function draws black circles around the dots (just to make it prettier).

```
library(PTCA4CATA)
PlotScree(ev = resCA.sym$ExPosition.Data$eigs,
          p.ev =  rescainf$Inference.Data$components$p.vals,
          title = 'Drug useby Age data Set. Eigenvalues Inference',
          plotKaiser = TRUE
)
```

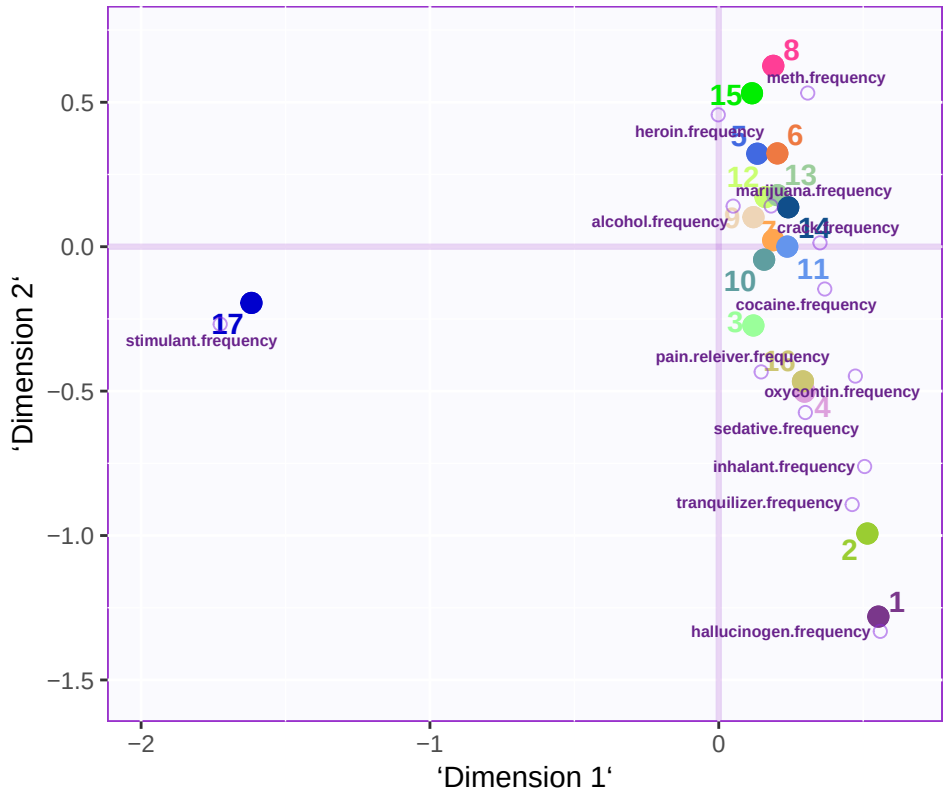


9.1 Factor Map for Symmetrical Graph

```
baseMap.i1 <- createFactorMap(Fi, constraints = constraints.sym,
                              col.points = color4drugs,
                             col.labels = color4drugs , cex = 5, text.cex = 4, pch = 20,
                             display.labels = TRUE , alpha.axes = 0.2,alpha.points = 1
)

print(baseMap.i1$zeMap + baseMap.i1$zeMap_dots)
```

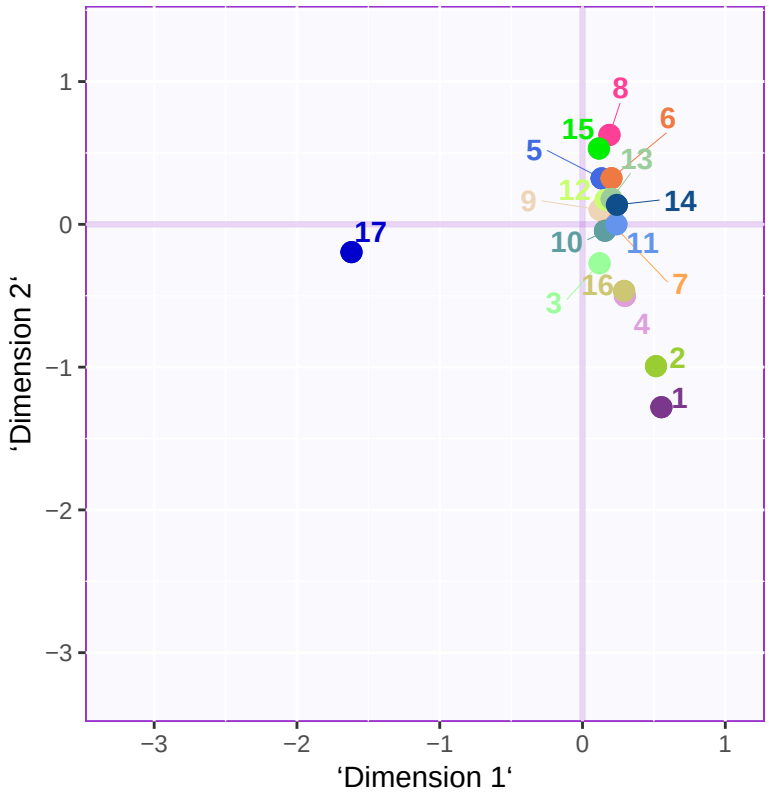



```
baseMap.j1 <- createFactorMap(Fj, constraints = constraints.sym,
                              color.points = "lightgreen",text.cex = 2,cex = 2,pch = 21)
print(baseMap.i1$zeMap + baseMap.i1$zeMap_dots + baseMap.j1$zeMap_dots + baseMap.j1$zeMap_text)
```

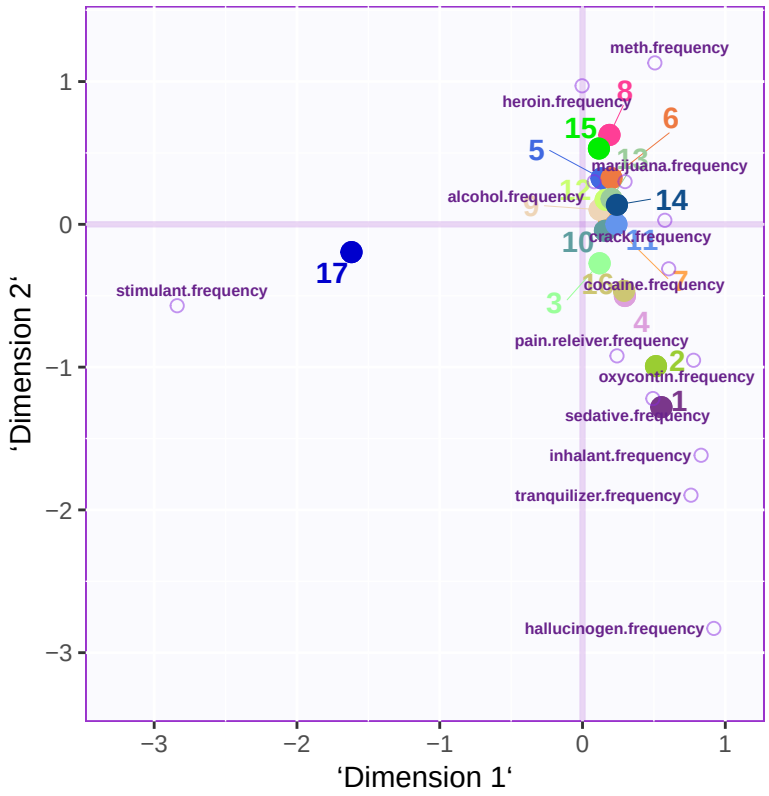


9.2 Factor Map for Asymmetrical Graph

```
baseMap.i2 <- createFactorMap(Fi, constraints = constraints.asym,
                              col.points = color4drugs,
                             col.labels = color4drugs , cex = 5, text.cex = 4, pch = 20,
                             display.labels = TRUE , alpha.axes = 0.2,alpha.points = 1
)
print(baseMap.i2$zeMap + baseMap.i2$zeMap_dots )
```

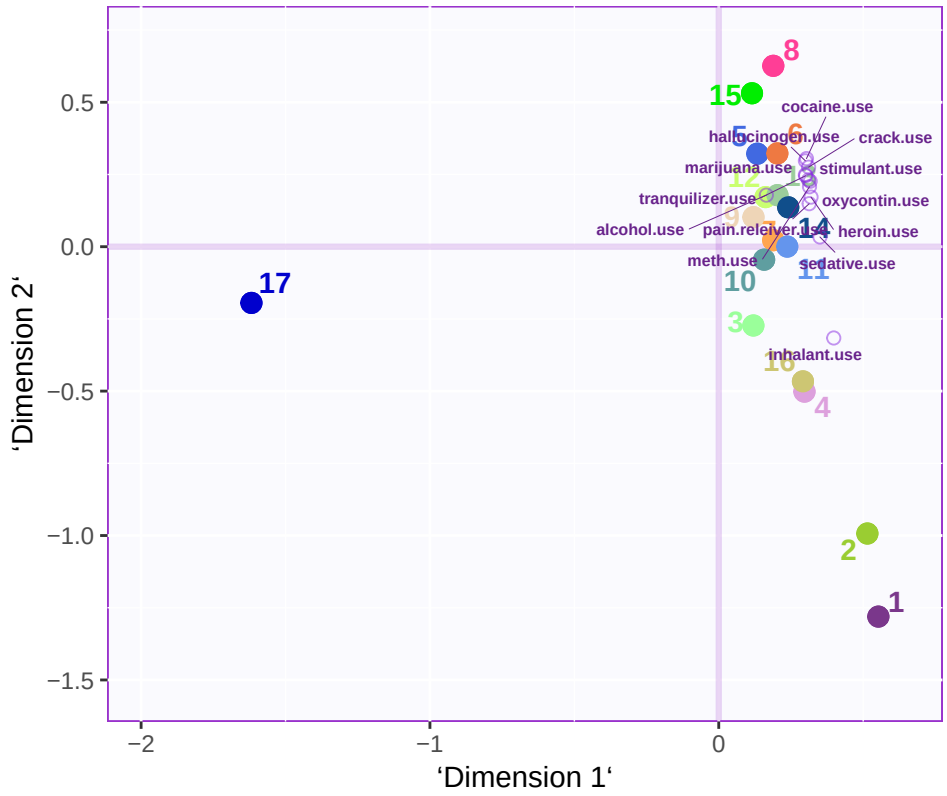



```
baseMap.j2 <- createFactorMap(Fj.a, constraints = constraints.asym,
                              color.points = "lightgreen",text.cex = 2,cex = 2,pch = 21)

print(baseMap.i2$zeMap + baseMap.i2$zeMap_dots+ baseMap.j2$zeMap_dots + baseMap.j2$zeMap_text)
```



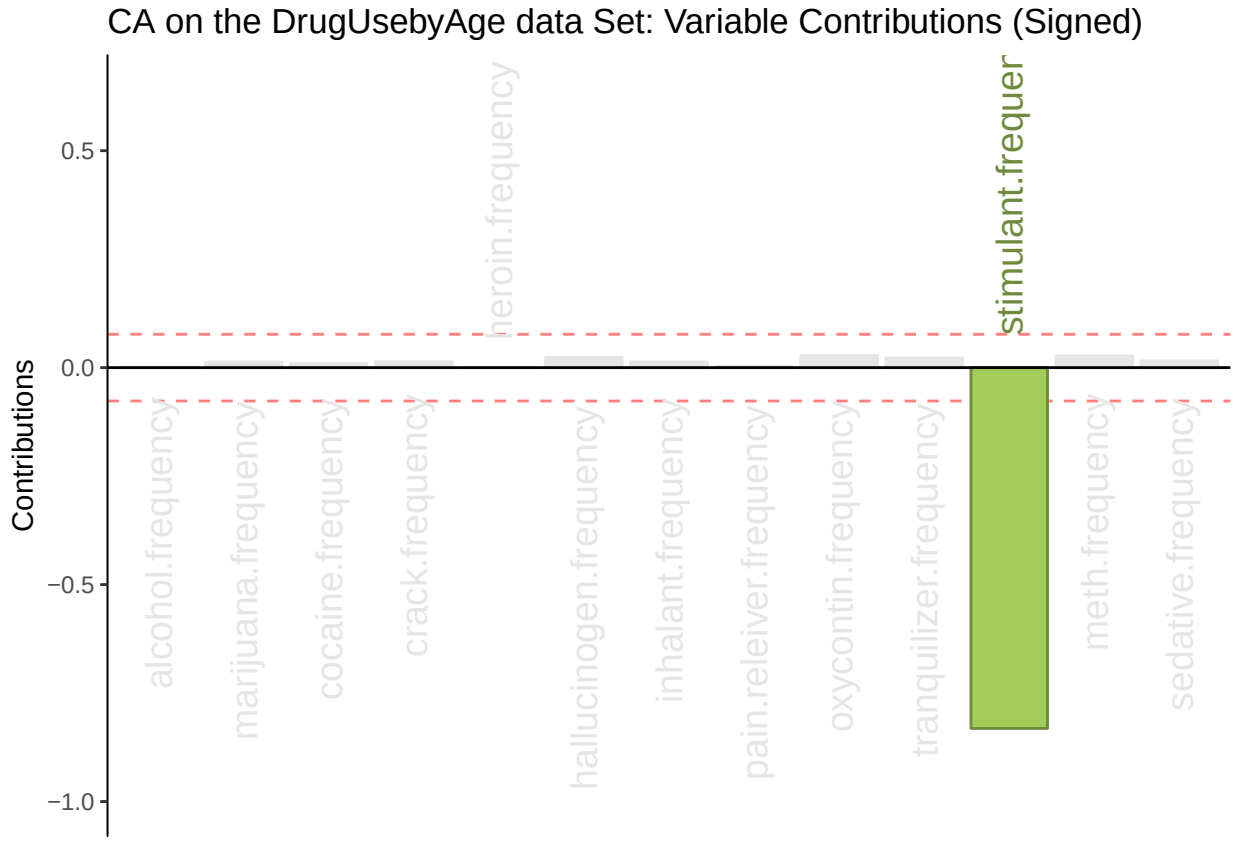
Factor Map for Supplementary Observations

```
HA.sup <- supplementaryRows(SUP.DATA = datasupp, res = resCA.sym)
HA.sup1 <- supplementaryCols(SUP.DATA = datasupp, res = resCA.sym)
baseMap.i11 <- createFactorMap(HA.sup1$fjj, constraints = constraints.sup,
                               color.points = "lightgreen",text.cex = 2,cex = 2,pch = 21)
print(baseMap.i1$zeMap + baseMap.i1$zeMap_dots + baseMap.i11$zeMap_dots + baseMap.i11$zeMap_text)
```

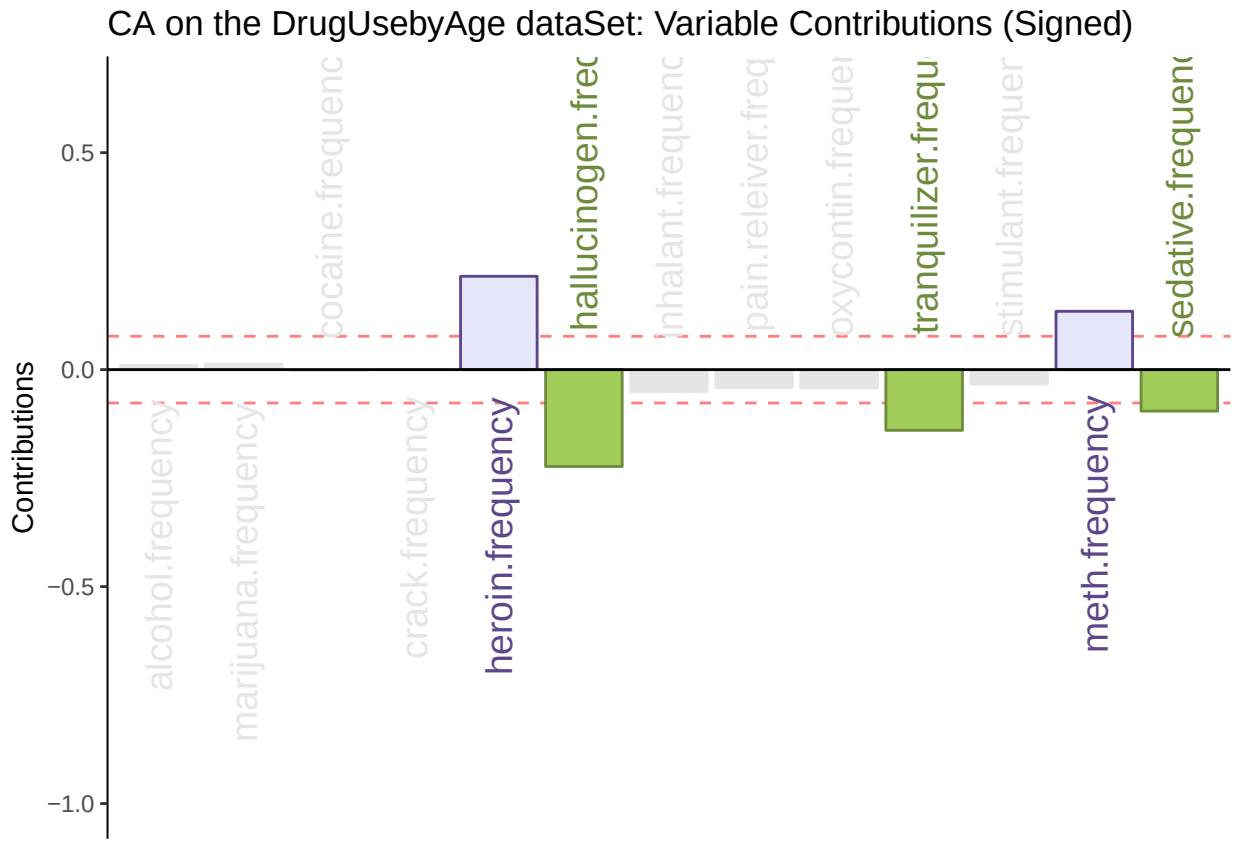


9.3 Contribution Bars for variables

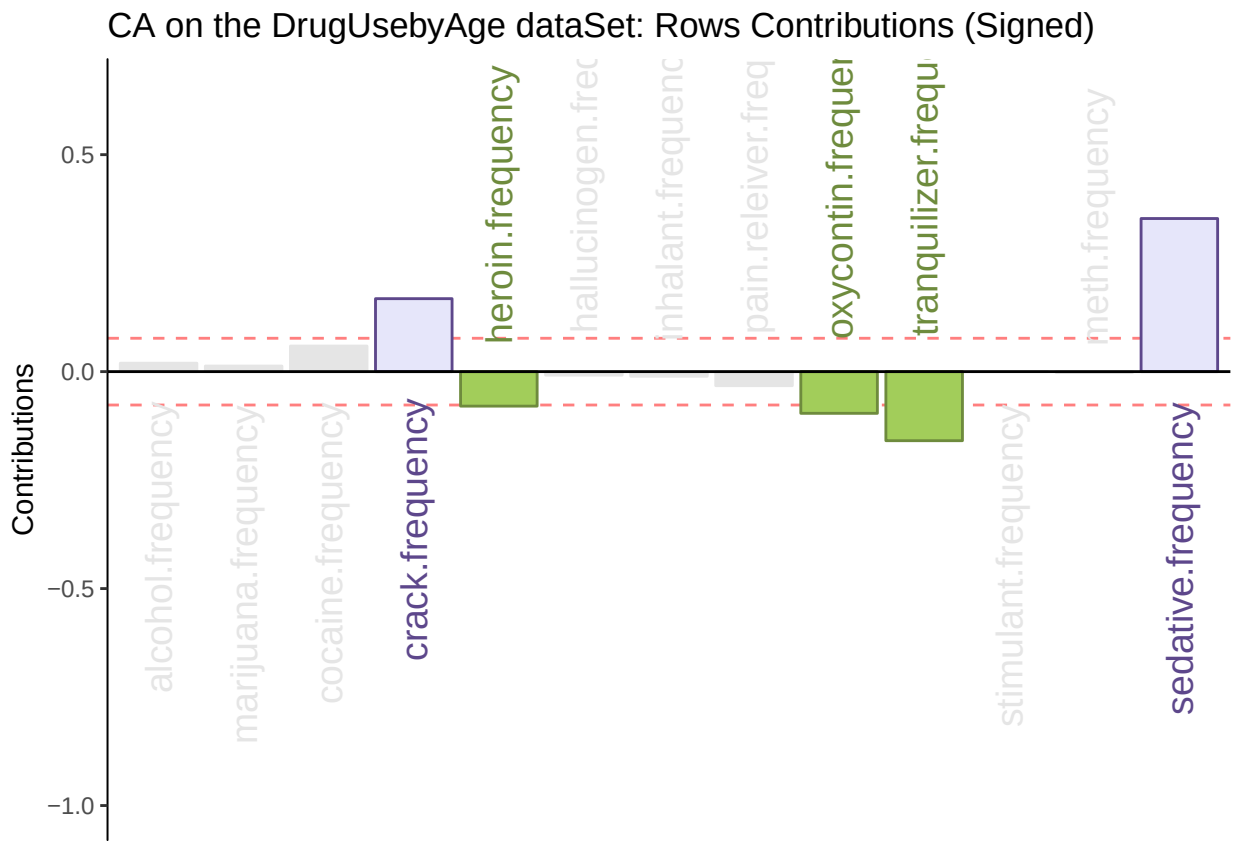
```
signed.ctrJ1 <- resCA.sym$ExPosition.Data$cj * sign(resCA.sym$ExPosition.Data$fj)
b003.ctrJ.s.11 <- PrettyBarPlot2(signed.ctrJ1[,1],
                                 threshold = 1 / NROW(signed.ctrJ1),
                                 font.size = 5,
                                 #color4bar = gplots::col2hex(color4drugs), # we need hex code
                                 main = 'CA on the DrugUsebyAge data Set: Variable Contributions (Signed
                                 ylab = 'Contributions',
                                 ylim = c(1.2*min(signed.ctrJ1), 1.2*max(signed.ctrJ1))
)
print(b003.ctrJ.s.11)
```




```
b004.ctrJ.s.21 <- PrettyBarPlot2(signed.ctrJ1[,2],
                                 threshold = 1 / NROW(signed.ctrJ1),
                                 font.size = 5,
                                 #color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                 main = 'CA on the DrugUsebyAge dataSet: Variable Contributions (Signed)',
                                 ylab = 'Contributions',
                                 ylim = c(1.2*min(signed.ctrJ1), 1.2*max(signed.ctrJ1))
)
print(b004.ctrJ.s.21)
```



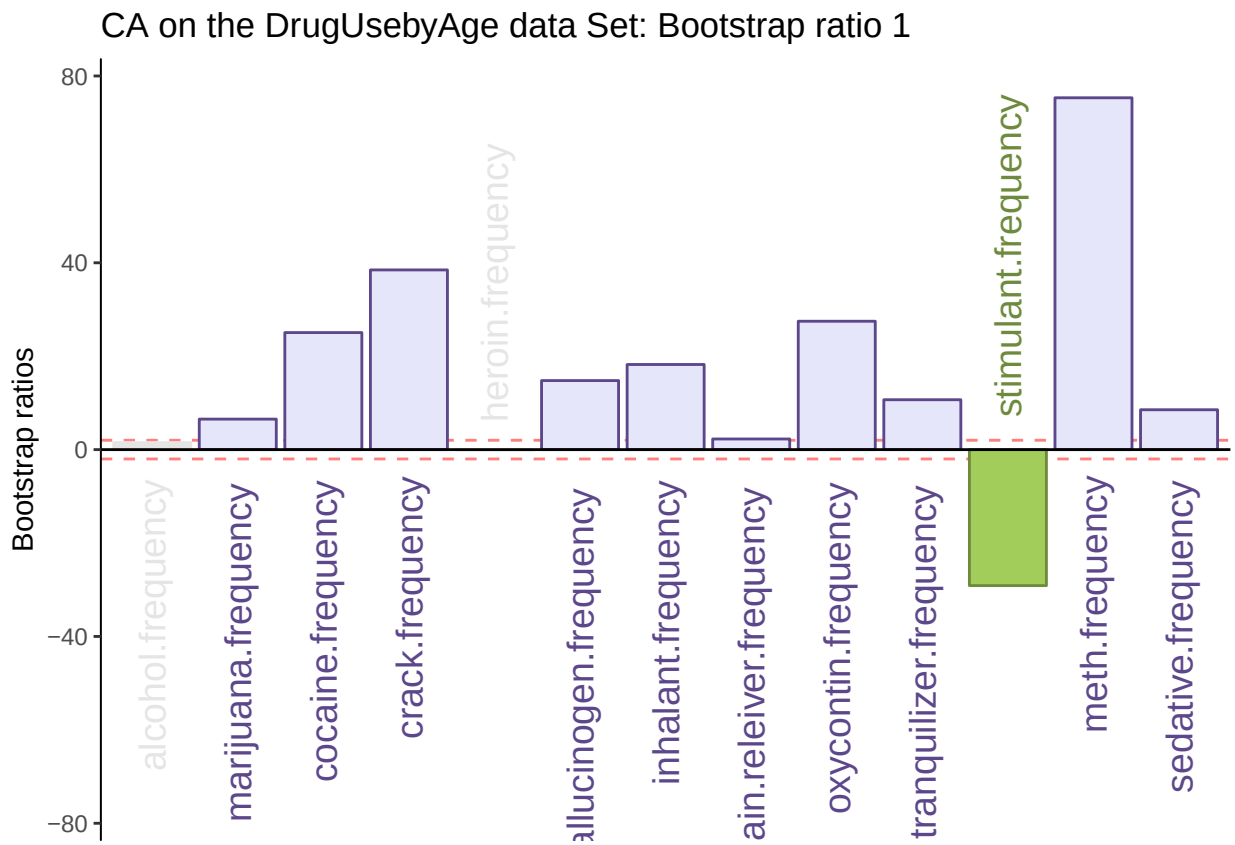

```
b004.ctrJ.s.31 <- PrettyBarPlot2(signed.ctrJ1[,3],
                                 threshold = 1 / NROW(signed.ctrJ1),
                                 font.size = 5,
                                #color4bar = gplots::col2hex(col4J.ibm), # we need hex code
                                 main = 'CA on the DrugUsebyAge dataSet: Rows Contributions (Signed)',
                                 ylab = 'Contributions',
                                 ylim = c(1.2*min(signed.ctrJ1), 1.2*max(signed.ctrJ1))
)
print(b004.ctrJ.s.31)
```



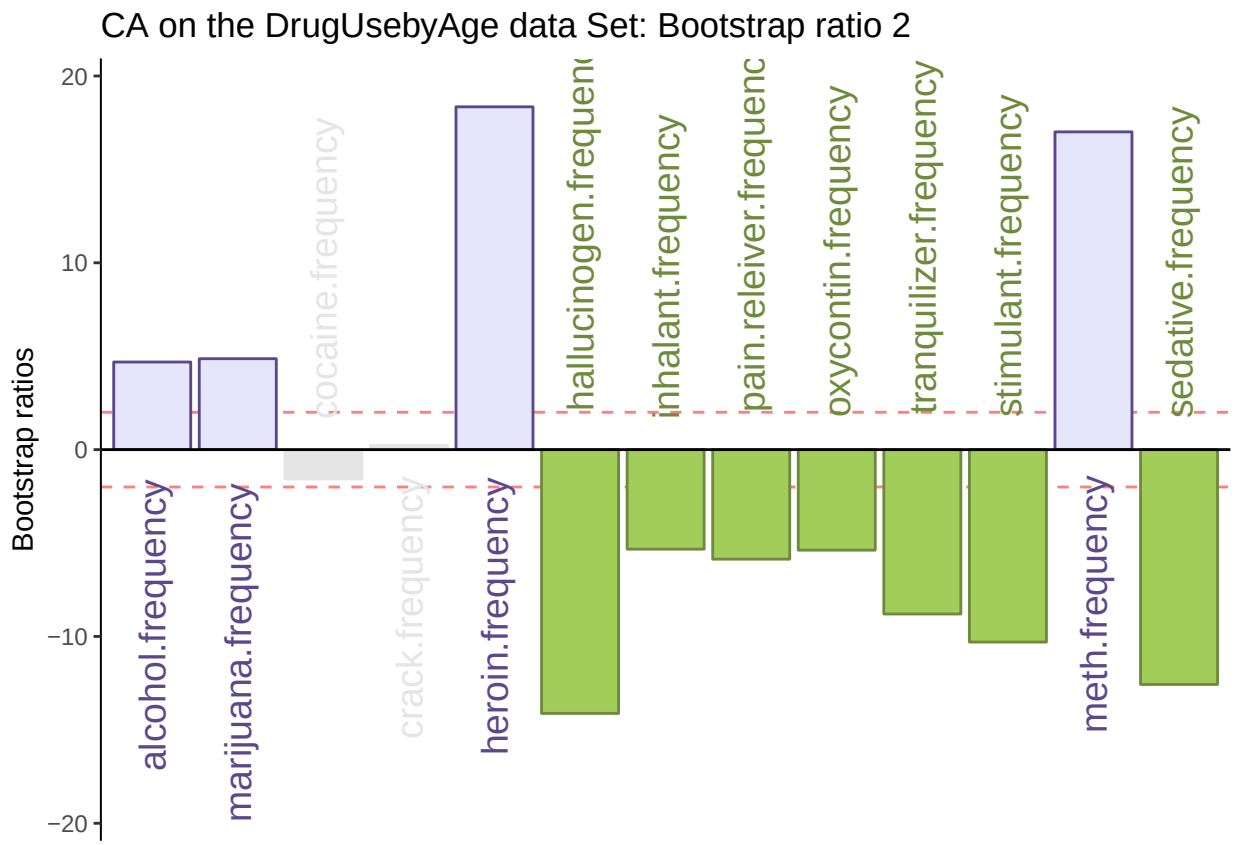
9.4 Bootstrap Ratios for variables

For the bootstrap ratios we can say that those having B.R greater than 2 are significant.

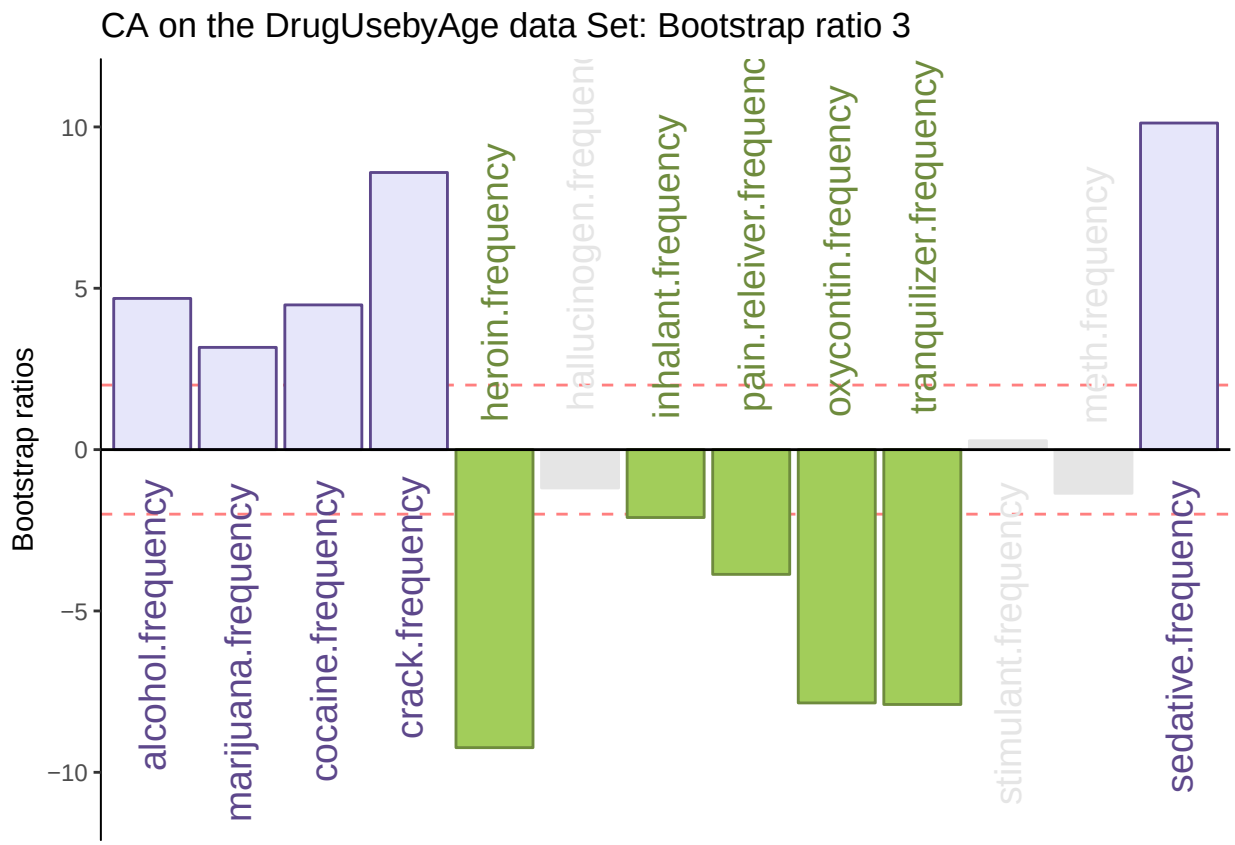
```
BR1 <- rescainf$Inference.Data$fj.boots$tests$boot.ratios
laDim1 = 1
ba001.BR11 <- PrettyBarPlot2(BR1[,laDim1],
                             threshold = 2,
                             font.size = 5,
                             #color4bar = gplots::col2hex(col4J.ibm),
                             main = paste0( 'CA on the DrugUsebyAge data Set: Bootstrap ratio ',laDim1),
                             ylab = 'Bootstrap ratios'
                             #ylim = c(1.2*min(BR[,laDim]), 1.2*max(BR[,laDim]))
)
print(ba001.BR11)
```




```
#
laDim2 = 2
ba002.BR21 <- PrettyBarPlot2(BR1[,laDim2],
                             threshold = 2,
                             font.size = 5,
                             #color4bar = gplots::col2hex(col4J.ibm),
                             main = paste0(
                               'CA on the DrugUsebyAge data Set: Bootstrap ratio ',laDim2),
                             ylab = 'Bootstrap ratios'
)
print(ba002.BR21)
```




```
laDim3 = 3
ba002.BR31 <- PrettyBarPlot2(BR1[,laDim3],
                             threshold = 2,
                             font.size = 5,
                             main = paste0(
                               'CA on the DrugUsebyAge data Set: Bootstrap ratio ',laDim3),
                             ylab = 'Bootstrap ratios'
)
print(ba002.BR31)
```



9.5 Summary

From the above Data Analysis we can infer that :

From the symmetrical plot, we can say that - Old aged people above the age of 65years are generally heavy on stimulants than the average use

- Kids around the age of 12 were consuming more hallucinogen drug than the average use
- Teenagers around the age of 13 were consuming more tranquilizer drug than the average use
- Teenagers around the age of 14 were consuming more sedative drug than the average use
- Teenagers around the age of 16 were consuming more heroin drug than the average use
- Teenagers around the age of 17 were consuming more heroin drug than the average use
- Youngsters around the age of 20 were consuming more alcholol and marijuana than the average use
- People around the age of 35-49 were consuming more heroin and meth drug than the average use
- People around the age of 22-23 were consuming more cocaine and crack drug than the average use
- Youngsters/People around the age of 24-34 were consuming more marijuana than the average use
- People around the 65+ age were consuming more stimulant than the average use

```
options(knitr.duplicate.label = 'allow')
```

10 DiSTATIS

DiSTATIS is a procedure that combines bootstrap estimation (to estimate the variability of the experimental conditions) and a new 3-way extension of MDS, that can be used to integrate the distance matrices generated by the bootstrap procedure and to represent the results as MDS-like maps. Reliability estimates are expressed as tolerance intervals which reflect the accuracy of the assignment of scans to experimental categories and as confidence intervals which generalize standard hypothesis testing.

The purpose of this study is to determine how musically trained and untrained listeners sort Western classical melodies into clusters based on perceived similarities. Listeners at three expertise levels sorted MIDI and natural excerpts from the piano music of Bach, Mozart, and Beethoven. We analyzed the data using DISTATIS, which showed an effect of composer with both MIDI and natural stimuli, and an effect of pianist with natural stimuli. However, there was only a weak effect of music training.

Task Participants sorted excerpts into three clusters according to whether a single composer could have written the pieces in the group. Participants could listen to each excerpt as many times as they wanted to.We investigated the effects of composer, pianist, and music training on sorting.To analyze the data, we applied DiSTATIS, a recent adaptation of multi-dimensional scaling specifically adapted to reveal the perceived dissimilarity among items, as well as to investigate group differences.

DataSet

Piano music from Bach, Mozart, Beethoven 36 excerpts from CD recordings, with 3 from each composer by each of 4 pianists: Arrau, Barenboim, Pires, Richter - Excerpts were 9 to 15 s long We presented the stimuli as audio icons arranged randomly on a PowerPoint slide.

```
for (i in 1:length(Design_Data)) {
  Design_Data[i] <- if (Design_Data[i] <= 1) 1 else if (Design_Data[i] <= 4 & Design_Data[i] >1) 2 else
}
```

We cateogrize the level of expertise as follows: Musicians a) musical training = 5 years and above N = 10 Moderate Musicians b) musical training = 1 to 4 years N = 17 Nonmusicians c) musical training = less than 1 year N = 10

```
color4mus <- c(1, 1, 2 ,2, 1 ,1 ,1 ,1, 1, 1 ,1, 1, 3, 2 ,3, 2, 3 ,1, 3 ,1, 2 ,2 ,3 ,2 ,2 ,1 ,1 ,1, 2 ,1
Judges <- paste0(Design_Data,1:length(Design_Data))
```

Create the set of distance matrices

```
DistanceCube <- DistatisR::DistanceFromSort(Sorting_Data)
```

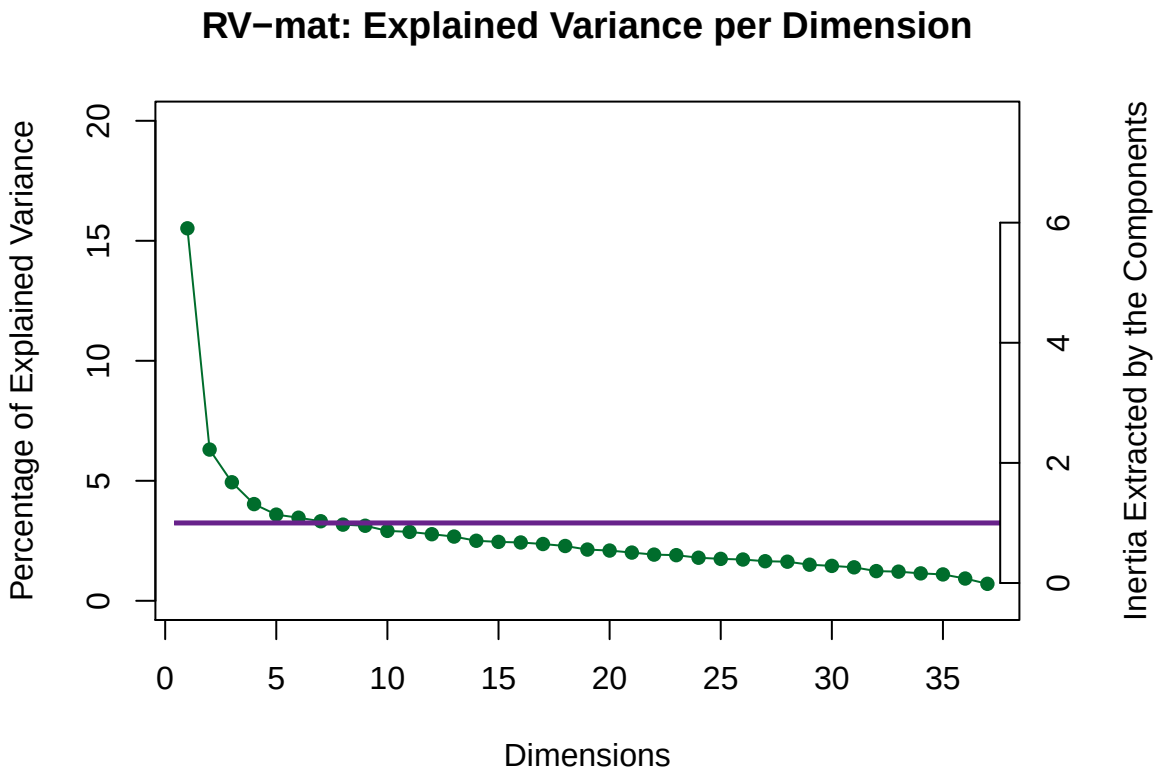
```
testDistatis <- DistatisR::distatis(DistanceCube)
```

```
BootF <- BootFactorScores(testDistatis$res4Splus$PartialF)
```

```
## [1] Bootstrap On Factor Scores. Iterations #:
## [2] 1000
```

ScreePlot

```
ev4C <- testDistatis$res4Cmat$eigValues
Scree.1 <-  PlotScree(ev = ev4C,
          p.ev = NULL, max.ev = NULL, alpha = 0.05,
          col.ns = "#006D2C", col.sig = "#54278F",
          title = "RV-mat: Explained Variance per Dimension",plotKaiser = TRUE)
```



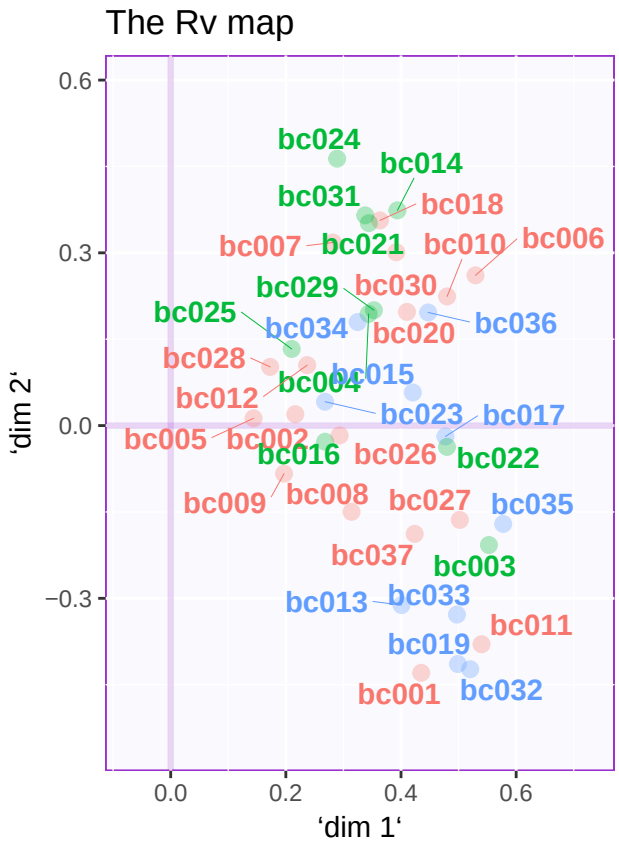
10.1 The Assessor Matrix

```
# Plot the assessor matrix
G <- testDistatis$res4Cmat$G
# Create a color scheme for the Composers
col4Non_musicions <- "#F8766D"
col4Moderate_musicions <- "#00BA38"
col4_musicions <- "#619CFF"
Judges[color4mus == 1 ] <- rep(col4Non_musicions,length(Judges))
```

```
## Warning in Judges[color4mus == 1] <- rep(col4Non_musicions,
## length(Judges)): number of items to replace is not a multiple of
## replacement length
```

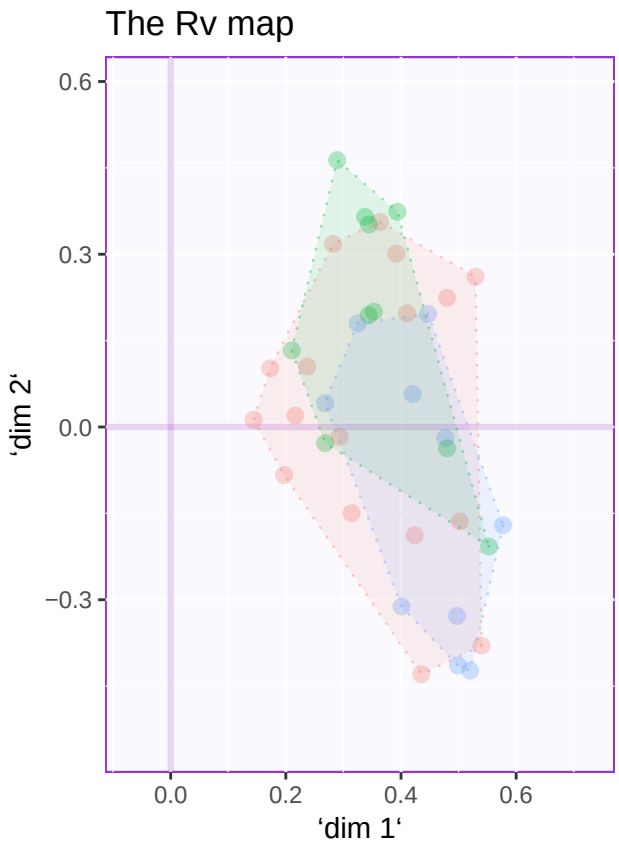
```
Judges[color4mus == 2 ] <- col4Moderate_musicions
Judges[color4mus == 3 ] <- col4_musicions
#------------------------------------------------------------------------
#------------------------------------------------------------------------
# A graph for the Judges set
baseMap.j <- PTCA4CATA::createFactorMap(G,
                                        title = 'The Rv map',
                                        col.points   = Judges,
                                        alpha.points =  .3,
                                        col.labels = Judges)
# A graph for the J-set
```

```
aggMap.j <- baseMap.j$zeMap_background + # background layer
            baseMap.j$zeMap_dots + baseMap.j$zeMap_text # dots & labels
# We print this Map with the following code
print(aggMap.j)
```



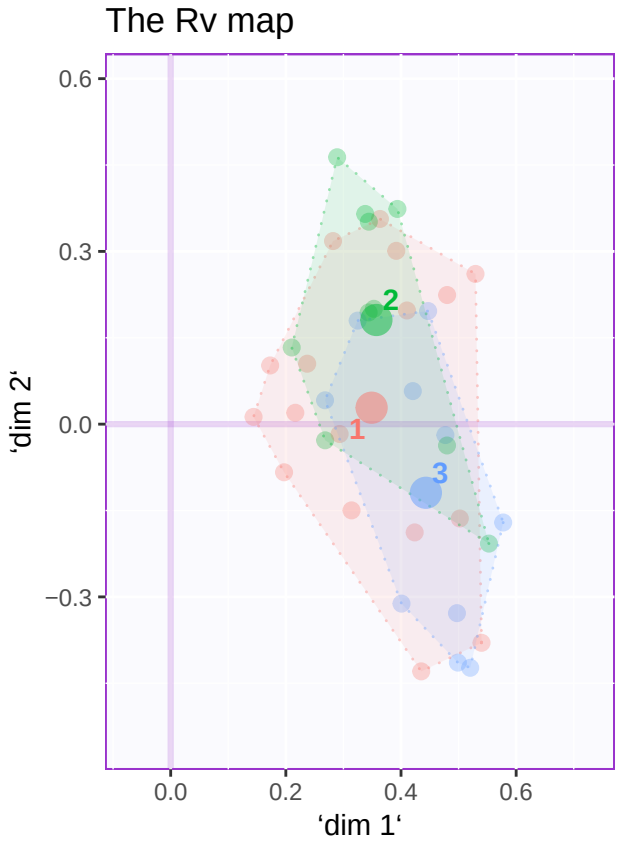

```
# Create 100% Tolerance interval polygons
#
GraphTJ.Hull.100 <- MakeToleranceIntervals(G,
                                    as.factor(Design_Data),
                                    names.of.factors = c("Dim1","Dim2"),
                                    col = unique(Judges),
                                            line.size = .5,
                                            line.type = 3,
                                            alpha.ellipse = .1,
                                            alpha.line = .4,
                                            p.level = 1, # full Hulls
                                            type = 'hull' #
                                            # use 'hull' for convex hull
)
#-----------------------------------------------------------------------
# Create the map
aggMap.j.withHull <- baseMap.j$zeMap_background + # background layer
                             baseMap.j$zeMap_dots + GraphTJ.Hull.100
#-----------------------------------------------------------------------
#-----------------------------------------------------------------------
# Plot it!
```

```
print(aggMap.j.withHull)
```

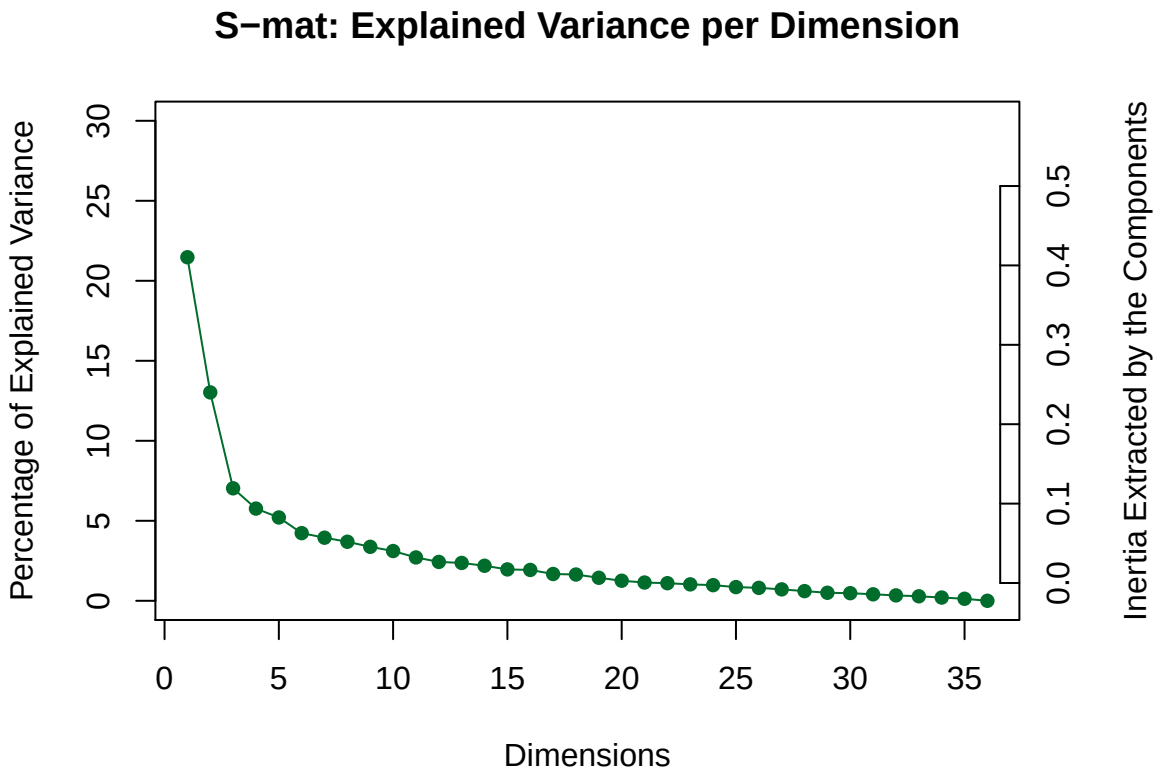



```
JudgesMeans.tmp <- aggregate(G, list(as.numeric(Design_Data)), mean) # compute the means
JudgesMeans <- JudgesMeans.tmp[,2:ncol(JudgesMeans.tmp )] # drop var 1
rownames(JudgesMeans) <- JudgesMeans.tmp[,1] # use var 1 to name the groups
#-------------------------------------------------------------------------
# a vector of color for the means
col4Means <- unique(Judges)
#-------------------------------------------------------------------------
# create the map for the means
MapGroup    <- PTCA4CATA::createFactorMap(JudgesMeans,
                                  axis1 = 1, axis2 = 2,
                                  constraints = baseMap.j$constraints,
                                          title = NULL,
                                          col.points = col4Means,
                                          display.points = TRUE,
                                          pch = 19, cex = 5,
                                          display.labels = TRUE,
                                          col.labels = col4Means,
                                          text.cex = 4,
                                          font.face = "bold",
                                          font.family = "sans",
                                          col.axes = "darkorchid",
                                          alpha.axes = 0.2,
                                          width.axes = 1.1,
                             col.background = adjustcolor("lavender",
```

```
                                          alpha.f = 0.2),
                                        force = 1, segment.size = 0)
# The map with observations and group means
aggMap.j.withMeans <- aggMap.j.withHull +
                    MapGroup$zeMap_dots + MapGroup$zeMap_text
#-----------------------------------------------------------------------
# plot it!
print(aggMap.j.withMeans)
```




```
# First we fix a bit of shamefull absentmindness:
#   The eigenvalues of the compromise matrix are not available
#   in DistatisR.
#   So we recompute them here
ev4S <- eigen(testDistatis$res4Splus$Splus,
              symmetric = TRUE, only.values = TRUE)$values
# A scree for the compromise
Scree.S <-  PlotScree(ev = ev4S,
                  p.ev = NULL, max.ev = NULL, alpha = 0.05,
                  col.ns = "#006D2C", col.sig = "#54278F",
                  title = "S-mat: Explained Variance per Dimension")
```

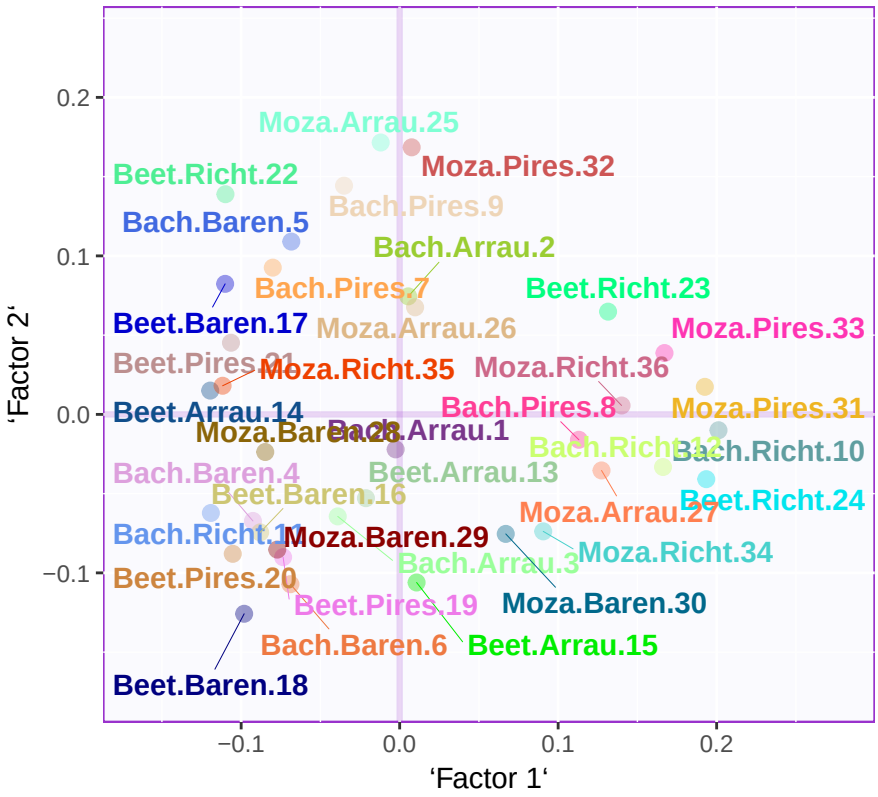



```
zeScree.S <- recordPlot()
```

10.2 I-Map

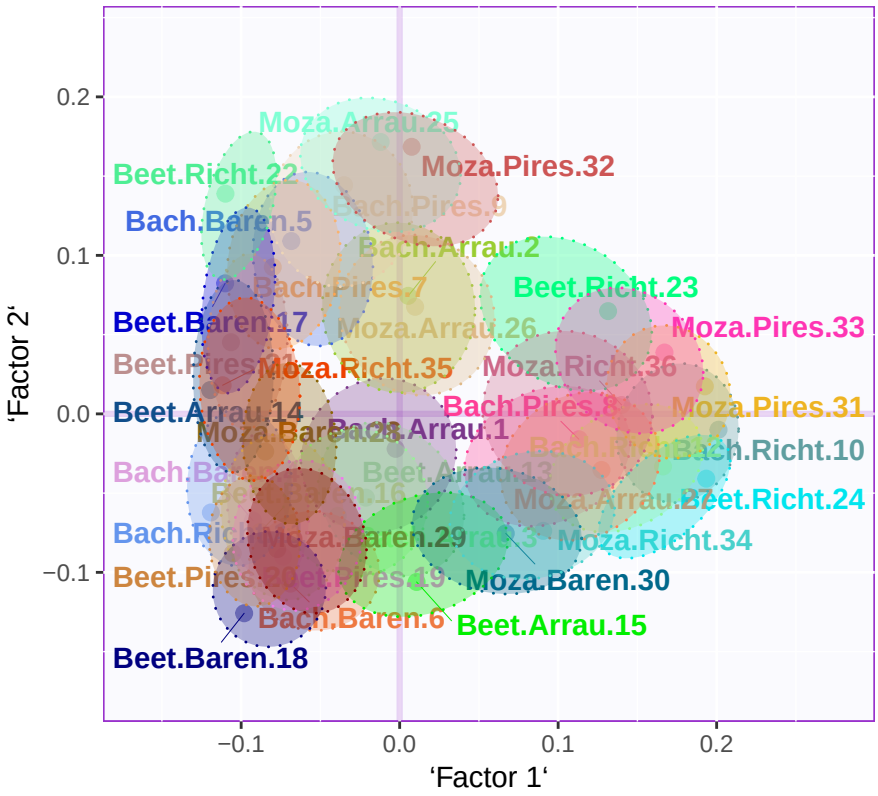
```
Fi <- testDistatis$res4Splus$F
col4Beers <- prettyGraphsColorSelection(nrow(Fi))
#  Use colors from prettyGraphs
#------------------------------------------------------------------------
# Graphs for the I set
#------------------------------------------------------------------------
# Create the base map
constraints4Fi <- lapply(minmaxHelper(Fi),'*',1.2)
baseMap.i <- PTCA4CATA::createFactorMap(Fi,
                                        col.points = col4Beers,
                                        col.labels = col4Beers,
                                        constraints = constraints4Fi,
                                        alpha.points =  .4)
#------------------------------------------------------------------------
# We are  interested about the labels here
# so we will use dots and labels
#------------------------------------------------------------------------
# Plain map with color for the I-set
aggMap.i <- baseMap.i$zeMap_background + baseMap.i$zeMap_dots +
                                         baseMap.i$zeMap_text
#------------------------------------------------------------------------
```

```
# print this Map
print(aggMap.i)
```




```
# Create Confidence Interval Plots
# use function MakeCIEllipses from package PTCA4CATA
#
constraints4Fi <- lapply(minmaxHelper(Fi),'*',1.2)
GraphElli <- MakeCIEllipses(BootF[,1:2,],
                      names.of.factors = c("Factor 1","Factor 2"),
                      alpha.line = .5,
                      alpha.ellipse = .3,
                      line.size = .5,
                      line.type = 3,
                      col = col4Beers,
                      p.level = .95 )
#-----------------------------------------------------------------------
# create the I-map with Observations and their confidence intervals
#
aggMap.i.withCI <-  aggMap.i +  GraphElli + MapGroup$zeMap_text
#-----------------------------------------------------------------------
# plot it!
print(aggMap.i.withCI)
```

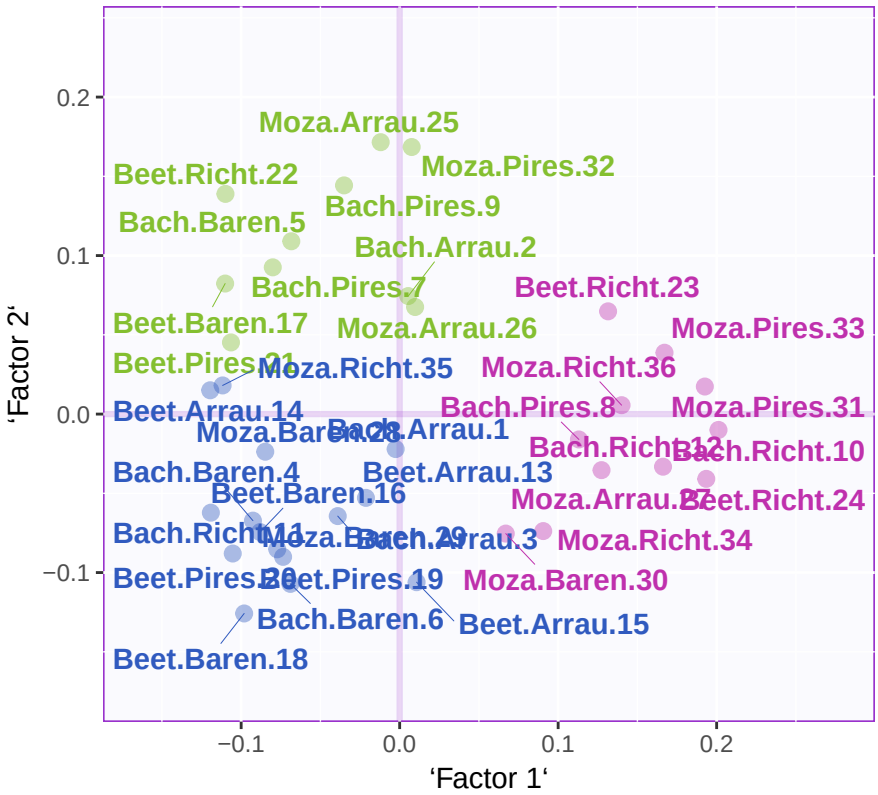
```
## Warning: Removed 3 rows containing missing values (geom_text_repel).
```




```
# Old graph with links to partial factor scores
# Not that informative for sorting tasks
# Change names of the assessors
partF <- testDistatis$res4Splus$PartialF
dimnames(partF)[[3]] <- as.character(1:dim(partF)[3])
PartialF <- GraphDistatisPartial(FS = testDistatis$res4Splus$F,
                     PartialFS = partF,
                     axis1 = 1, axis2 = 2, constraints = NULL,
                     item.colors = col4Beers,
                     participant.colors = NULL,
                     ZeTitle = "Distatis-Partial",
                     Ctr=NULL, color.by.observations = TRUE,
                     nude = FALSE, lines = TRUE)
#Plot it !
F.and.PartialF <- recordPlot()
```

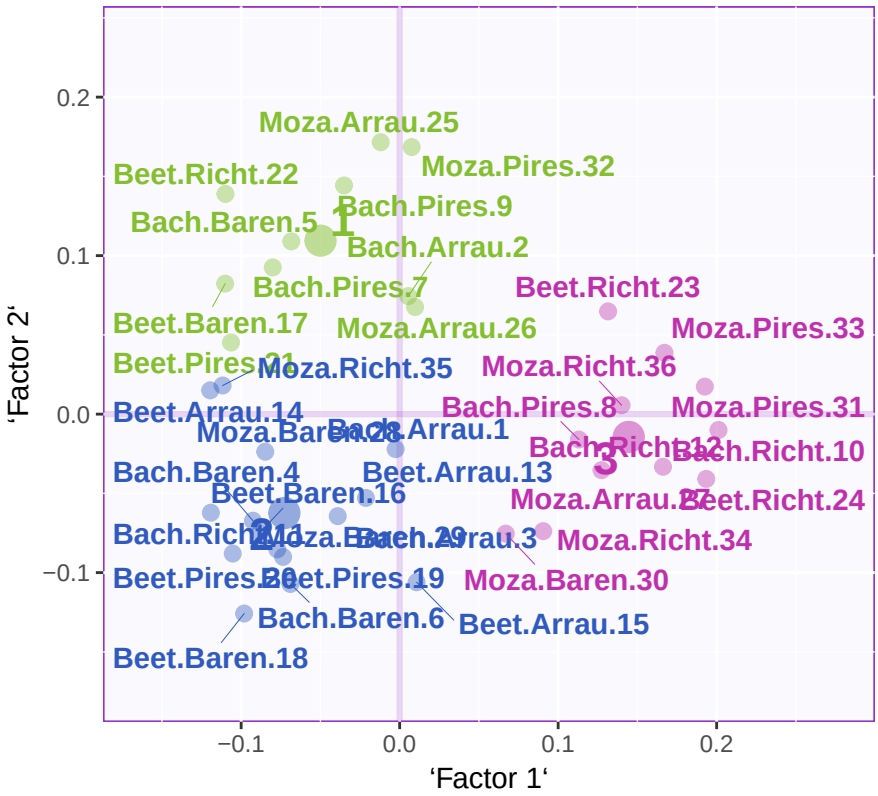
```
# Some classification now
# First plain k-means
set.seed(42)
beers.kMeans <- kmeans(x = Fi , centers = 3)
#------------------------------------------------------------------------
# Now to get a map by cluster:
col4Clusters  <- createColorVectorsByDesign(
               makeNominalData(
               as.data.frame(beers.kMeans$cluster)  ))
```

```
#=========================================================================
#-------------------------------------------------------------------------
# Graphs for the I set
#-------------------------------------------------------------------------
# Create the base map
# constraints4Fi <- lapply(minmaxHelper(Fi),'*',1.2)
baseMap.i.km <- PTCA4CATA::createFactorMap(Fi,
                                           col.points = col4Clusters$oc,
                                           col.labels = col4Clusters$oc,
                                           constraints = constraints4Fi,
                                           alpha.points =  .4)
#-------------------------------------------------------------------------
# We are  interested about the labels here
# so we will use dots and labels
#-------------------------------------------------------------------------
# Plain map with color for the I-set
aggMap.i.km <- baseMap.i.km$zeMap_background +
  baseMap.i.km$zeMap_dots + baseMap.i.km$zeMap_text
# print
print(aggMap.i.km)
```



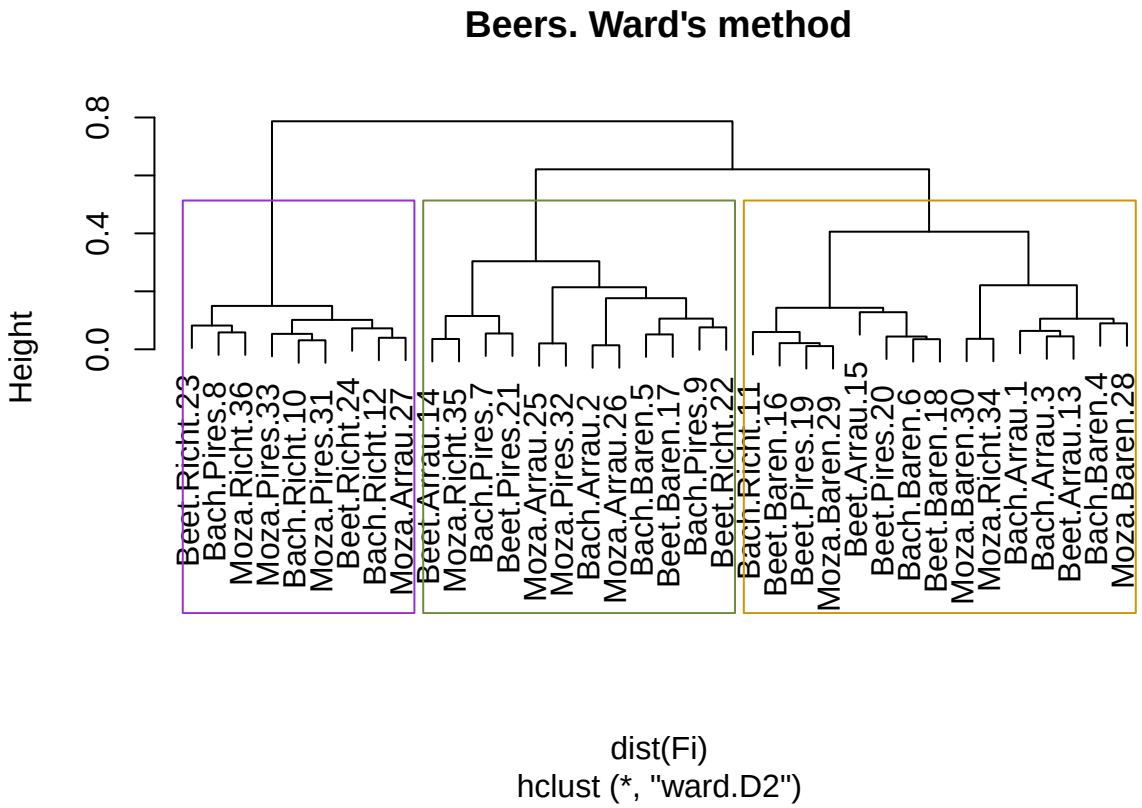

```
# get the color order in the c=good order
col4C <- col4Clusters$gc[sort(rownames(col4Clusters$gc),
                              index.return = TRUE)$ix]
# create the map for the means
map4Clusters    <- PTCA4CATA::createFactorMap(beers.kMeans$centers,
```

```
                                        axis1 = 1, axis2 = 2,
                                        constraints = constraints4Fi,
                                        title = NULL,
                                        col.points = col4C,
                                        display.points = TRUE,
                                        pch = 19, cex = 5,
                                        display.labels = TRUE,
                                        col.labels = col4C,
                                        text.cex = 6,
                                        font.face = "bold",
                                        font.family = "sans",
                                        col.axes = "darkorchid",
                                        alpha.axes = 0.2,
                                        width.axes = 1.1,
                                        col.background =
                              adjustcolor("lavender", alpha.f = 0.2),
                                        force = 1, segment.size = 0)
# The map with observations and group means
aggMap.i.withCenters <- aggMap.i.km  +
                map4Clusters$zeMap_dots + map4Clusters$zeMap_text
#
print(aggMap.i.withCenters)
```



10.3 Cluster Analysis

```
# A cluster analysis
beer.hc <- hclust(d = dist(Fi),
                  method = 'ward.D2' )

plot.tree <- plot(beer.hc,  main = "Beers. Ward's method")
hc.3.cl <- rect.hclust(beer.hc, k = 3,
               border = c('darkorchid',
                          'darkolivegreen4','darkgoldenrod3')
                         )
```




```
library(useful)
```

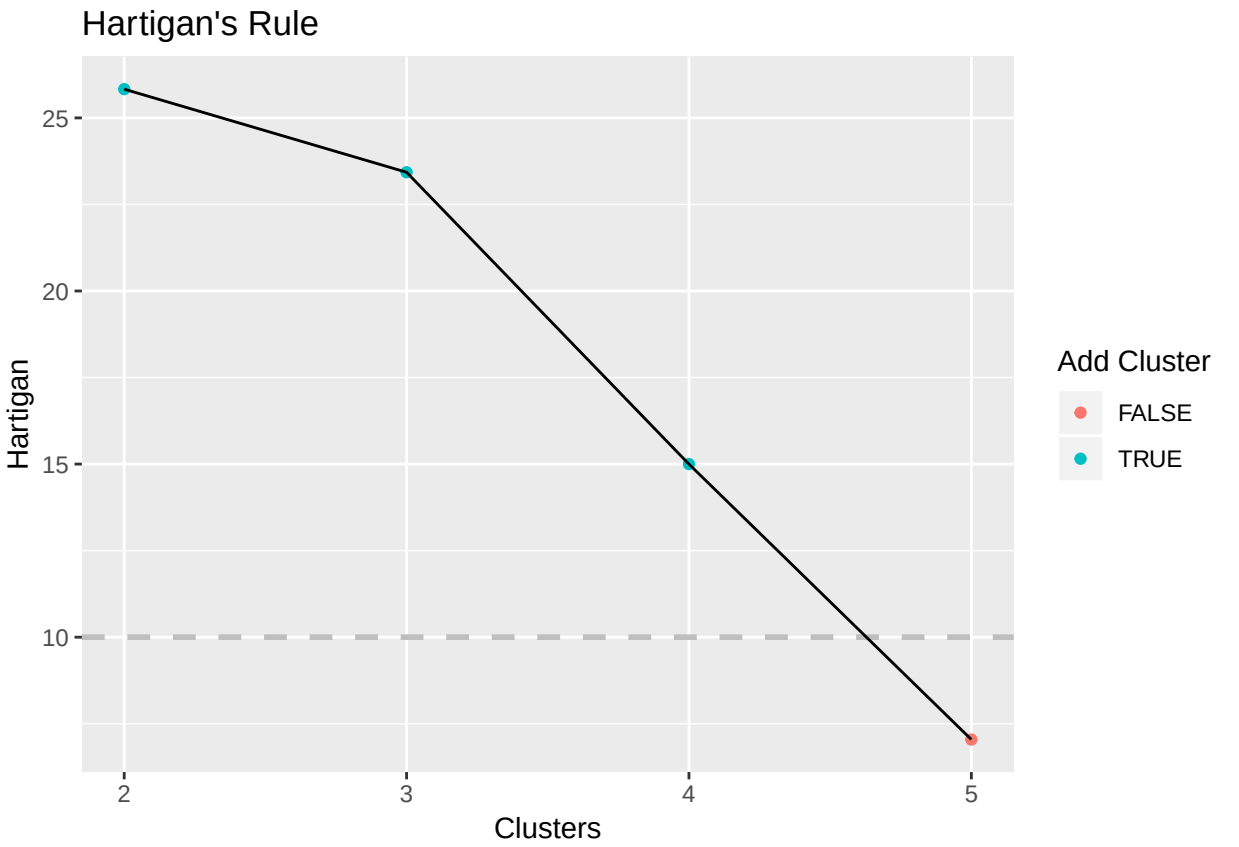
```
## Loading required package: ggplot2
```

```
best.beers <- useful::FitKMeans(Fi, max.clusters = 5,
                                seed = 314)
print(best.beers) # when Hartigan parameter > 10 => add a cluster
```

```
##   Clusters  Hartigan AddCluster
## 1        2 25.830051       TRUE
## 2        3 23.429672       TRUE
## 3        4 14.998705       TRUE
## 4        5  7.042848      FALSE
```

```
plot.harti <- useful::PlotHartigan(best.beers)
```

```
print(plot.harti)
```



From the Hartigan's Rule we can say that their are three cluster's in the data which was also the same as taken before.

10.4 Summary

Participants were able to strongly differentiate Mozarts excerpts from Beethovens, with Bach falling in between those two and Richters performances of the three composers were clustered relatively close to the Mozart region of the solution, indicating their clarity and balance; in contrast, those of Barenboim were clustered in the Beethoven region, indicating their sumptuousness and passion.