Factor Analysis for Customer Churn

Factor Analysis (FA) is an exploratory information examination strategy used to look through powerful fundamental elements or idle factors from a lot of watched factors. It helps in information translations by diminishing the quantity of factors. It extricates most extreme normal change from all factors and places them into a typical score.

Factor examination is generally used in statistical surveying, publicizing, brain research, account, and activity inquire about. Economic specialists use factor examination to recognize value delicate clients, distinguish brand includes that impact shopper decision, and aides in understanding channel determination criteria for the conveyance channel.

```
custc <-
read.csv("C:/Users/admin/Desktop/MVA/PROJECT/TelEco_Customer_Churn.csv")
dim(custc)

## [1] 7043   21

#structure of dataset
str(custc)

## 'data.frame':    7043 obs. of  21 variables:
##  $ customerID      : Factor w/ 7043 levels "0002-ORFBO","0003-MKNFE",..:
5376 3963 2565 5536 6512 6552 1003 4771 5605 4535 ...
##  $ gender          : Factor w/ 2 levels "Female","Male": 1 2 2 2 1 1 2 1 1
2 ...
##  $ SeniorCitizen   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Partner         : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 1 1 2 1
...
##  $ Dependents      : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 2 1 1 2
...
##  $ tenure          : int  1 34 2 45 2 8 22 10 28 62 ...
##  $ PhoneService    : Factor w/ 2 levels "No","Yes": 1 2 2 1 2 2 2 1 2 2
...
##  $ MultipleLines   : Factor w/ 3 levels "No","No phone service",..: 2 1 1
2 1 3 3 2 3 1 ...
##  $ InternetService : Factor w/ 3 levels "DSL","Fiber optic",..: 1 1 1 1 2
2 2 1 2 1 ...
##  $ OnlineSecurity  : Factor w/ 3 levels "No","No internet service",..: 1 3
3 3 1 1 1 3 1 3 ...
##  $ OnlineBackup    : Factor w/ 3 levels "No","No internet service",..: 3 1
3 1 1 1 3 1 1 3 ...
##  $ DeviceProtection: Factor w/ 3 levels "No","No internet service",..: 1 3
1 3 1 3 1 1 3 1 ...
##  $ TechSupport     : Factor w/ 3 levels "No","No internet service",..: 1 1
1 3 1 1 1 1 3 1 ...
##  $ StreamingTV     : Factor w/ 3 levels "No","No internet service",..: 1 1
1 1 1 3 3 1 3 1 ...
```

```
##  $ StreamingMovies : Factor w/ 3 levels "No","No internet service",..: 1 1
1 1 1 3 1 1 3 1 ...
##  $ Contract        : Factor w/ 3 levels "Month-to-month",..: 1 2 1 2 1 1 1
1 1 2 ...
##  $ PaperlessBilling: Factor w/ 2 levels "No","Yes": 2 1 2 1 2 2 2 1 2 1
...
##  $ PaymentMethod   : Factor w/ 4 levels "Bank transfer (automatic)",..: 3
4 4 1 3 3 2 4 3 1 ...
##  $ MonthlyCharges  : num  29.9 57 53.9 42.3 70.7 ...
##  $ TotalCharges    : num  29.9 1889.5 108.2 1840.8 151.7 ...
##  $ Churn           : Factor w/ 2 levels "No","Yes": 1 1 2 1 2 2 1 1 2 1
...
```

```r
sapply(custc, function(x) sum(is.na(x)))
```

```
##       customerID           gender    SeniorCitizen          Partner
##                0                0                0                0
##       Dependents           tenure     PhoneService    MultipleLines
##                0                0                0                0
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##                0                0                0                0
##      TechSupport      StreamingTV  StreamingMovies         Contract
##                0                0                0                0
## PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges
##                0                0                0               11
##            Churn
##                0
```

```r
#
custc <- custc[complete.cases(custc),]  ## to remove which has null values
sapply(custc, function(x) sum(is.na(x)))
```

```
##       customerID           gender    SeniorCitizen          Partner
##                0                0                0                0
##       Dependents           tenure     PhoneService    MultipleLines
##                0                0                0                0
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##                0                0                0                0
##      TechSupport      StreamingTV  StreamingMovies         Contract
##                0                0                0                0
## PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges
##                0                0                0                0
##            Churn
##                0
```

```r
dim(custc)
```

```
## [1] 7032   21
```

```r
quant_var_df<-
data.frame(custc$tenure,custc$MonthlyCharges,custc$TotalCharges)
quant_var_df
```

```
##      custc.tenure custc.MonthlyCharges custc.TotalCharges
## 1               1                29.85              29.85
## 2              34                56.95            1889.50
## 3               2                53.85             108.15
## 4              45                42.30            1840.75
## 5               2                70.70             151.65
## 6               8                99.65             820.50
## 7              22                89.10            1949.40
## 8              10                29.75             301.90
## 7009           39                20.15             826.00
## 7010           12                19.20             239.00
## 7011           12                59.80             727.80
## 7012           72               104.95            7544.30
## 7013           63               103.50            6479.40
## 7014           44                84.80            3626.35
## 7015           18                95.05            1679.40
## 7016            9                44.20             403.35
## 7017           13                73.35             931.55
## 7018           68                64.10            4326.25
## 7019            6                44.40             263.05
## 7020            2                20.05              39.25
## 7021           55                60.00            3316.10
## 7022            1                75.75              75.75
## 7023           38                69.50            2625.25
## 7024           67               102.95            6886.25
## 7025           19                78.70            1495.10
## 7026           12                60.65             743.30
## 7027           72                21.15            1419.40
## 7028           24                84.80            1990.50
## 7029           72               103.20            7362.90
## 7030           11                29.60             346.45
## 7031            4                74.40             306.60
## 7032           66               105.65            6844.50
```

```r
attach(quant_var_df)
```

```r
#install.packages("PerformanceAnalytics")
library(PerformanceAnalytics)
```

```
## Warning: package 'PerformanceAnalytics' was built under R version 3.6.3
```

```
## Loading required package: xts
```

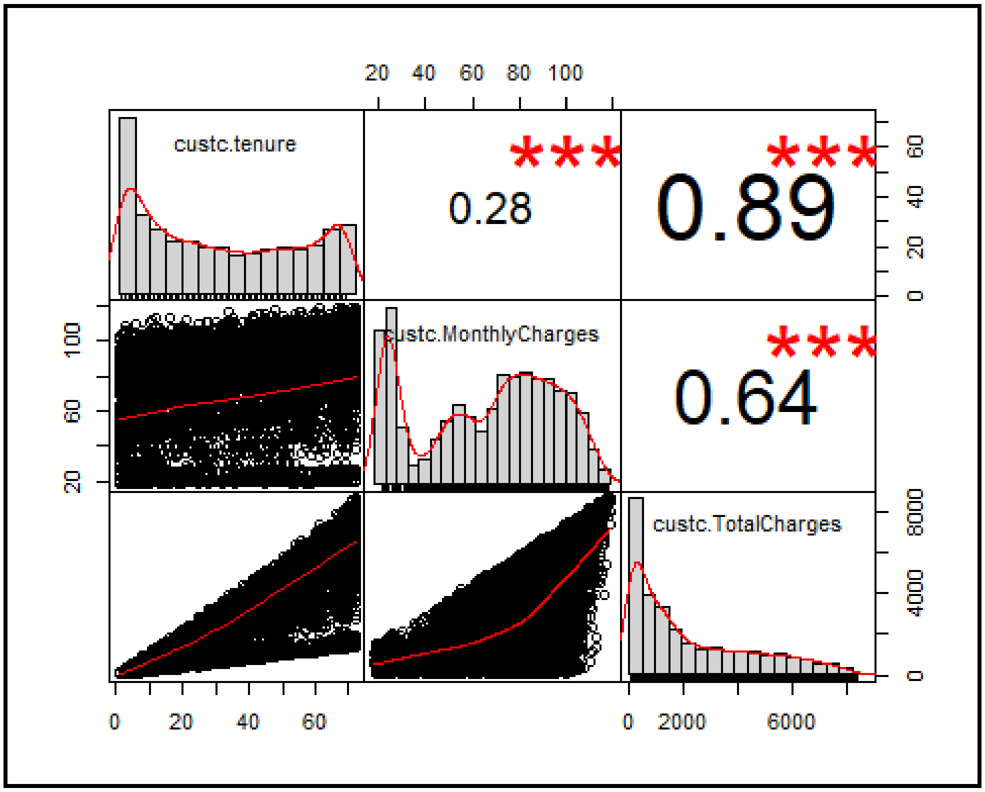
```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend
```

```r
chart.Correlation(quant_var_df, method = c("spearman"),histogram = TRUE, pch
= "19", cex= 0.7)
```

```
## Warning in cor.test.default(as.numeric(x), as.numeric(y), method =
method):
## Cannot compute exact p-value with ties

## Warning in cor.test.default(as.numeric(x), as.numeric(y), method =
method):
## Cannot compute exact p-value with ties

## Warning in cor.test.default(as.numeric(x), as.numeric(y), method =
method):
## Cannot compute exact p-value with ties
```

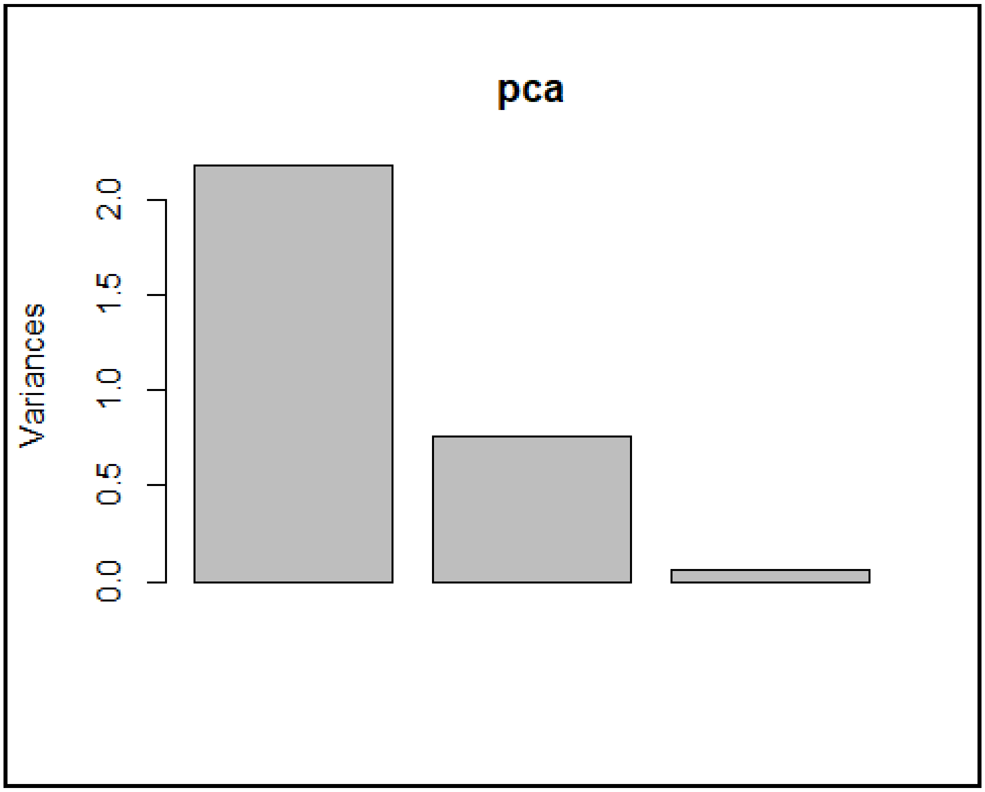



```
# apply PCA
pca<-prcomp(quant_var_df[,],scale=TRUE)
pca

## Standard deviations (1, .., p=3):
## [1] 1.4764138 0.8722163 0.2438052
##
## Rotation (n x k) = (3 x 3):
##                            PC1         PC2        PC3
## custc.tenure         0.5672112  0.60697524  0.5566440
## custc.MonthlyCharges 0.4857136 -0.79237469  0.3690862
## custc.TotalCharges   0.6650968  0.06101971 -0.7442600

# sample scores stored in pca$x
# singular values (square roots of eigenvalues) stored in pca$sdev
# loadings (eigenvectors) are stored in pca$rotation
# variable means stored in pca$center
# variable standard deviations stored in pca$scale
# A table containing eigenvalues and %'s accounted, follows
# Eigenvalues are sdev^2

(eigen_custc <- pca$sdev^2)

## [1] 2.1797978 0.7607612 0.0594410

names(eigen_custc) <- paste("PC",1:3,sep="")
eigen_custc
```

```
##       PC1       PC2       PC3
## 2.1797978 0.7607612 0.0594410

sumlambdas <- sum(eigen_custc)
sumlambdas

## [1] 3

propvar <- eigen_custc/sumlambdas
propvar

##        PC1        PC2        PC3
## 0.72659927 0.25358707 0.01981367

#pc1 holds 72 % of variance ,pc1 n pc2 holds 82% of variance

cumvar_custc <- cumsum(propvar)
cumvar_custc

##       PC1       PC2       PC3
## 0.7265993 0.9801863 1.0000000

matlambdas <- rbind(eigen_custc,propvar,cumvar_custc)
rownames(matlambdas) <- c("Eigenvalues","Prop. variance","Cum. prop. 
variance")
eigvec.custc <- pca$rotation
round(matlambdas,4)

##                       PC1    PC2    PC3
## Eigenvalues        2.1798 0.7608 0.0594
## Prop. variance     0.7266 0.2536 0.0198
## Cum. prop. variance 0.7266 0.9802 1.0000

summary(pca)

## Importance of components:
##                           PC1    PC2     PC3
## Standard deviation     1.4764 0.8722 0.24381
## Proportion of Variance 0.7266 0.2536 0.01981
## Cumulative Proportion  0.7266 0.9802 1.00000

pca$rotation

##                           PC1         PC2        PC3
## custc.tenure         0.5672112  0.60697524  0.5566440
## custc.MonthlyCharges 0.4857136 -0.79237469  0.3690862
## custc.TotalCharges   0.6650968  0.06101971 -0.7442600

print(pca)

## Standard deviations (1, .., p=3):
## [1] 1.4764138 0.8722163 0.2438052
##
```

```
## Rotation (n x k) = (3 x 3):
##                             PC1         PC2        PC3
## custc.tenure         0.5672112  0.60697524  0.5566440
## custc.MonthlyCharges 0.4857136 -0.79237469  0.3690862
## custc.TotalCharges   0.6650968  0.06101971 -0.7442600
```

```
plot(pca)
```




```
# Taking the first two PCs to generate linear combinations for all the
variables with two factors
pcafactors.custc <- eigvec.custc[,1:2]
pcafactors.custc
```

```
##                             PC1         PC2
## custc.tenure         0.5672112  0.60697524
## custc.MonthlyCharges 0.4857136 -0.79237469
## custc.TotalCharges   0.6650968  0.06101971
```

```
# Multiplying each column of the eigenvector's matrix by the square-root of
the corresponding eigenvalue in order to get the factor loadings
unrot.fact.custc <- sweep(pcafactors.custc,MARGIN=2,pca$sdev[1:2],`*`)
unrot.fact.custc
```

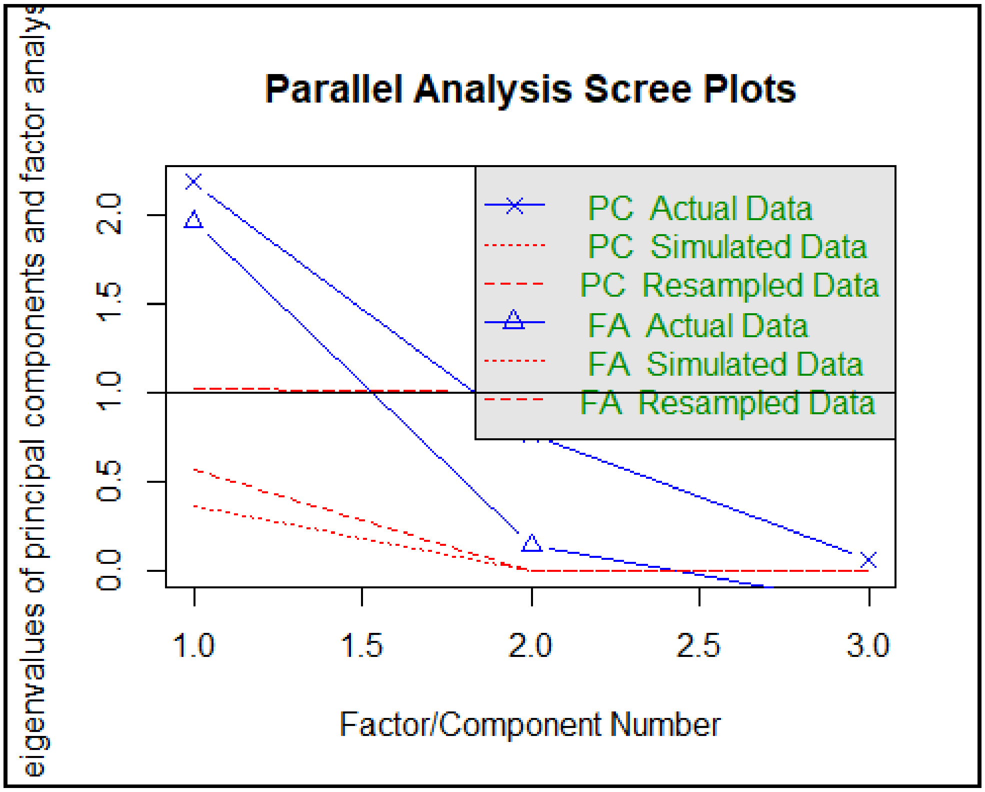
```
##                             PC1         PC2
## custc.tenure         0.8374385  0.52941367
```

```
## custc.MonthlyCharges 0.7171143 -0.69112209
## custc.TotalCharges   0.9819581  0.05322239

# Computing communalities
communalities.custc <- rowSums(unrot.fact.custc^2)
communalities.custc

##          custc.tenure custc.MonthlyCharges   custc.TotalCharges
##             0.9815821            0.9919027            0.9670743

# Performing the varimax rotation. The default in the varimax function is
norm=TRUE thus, Kaiser normalization is carried out
rot.fact.custc <- varimax(unrot.fact.custc)
View(unrot.fact.custc)
rot.fact.custc

## $loadings
##
## Loadings:
##                        PC1    PC2
## custc.tenure          0.988
## custc.MonthlyCharges  0.162 -0.983
## custc.TotalCharges    0.819 -0.544
##
##                    PC1   PC2
## SS loadings    1.674 1.267
## Proportion Var 0.558 0.422
## Cumulative Var 0.558 0.980
##
## $rotmat
##           [,1]       [,2]
## [1,] 0.8021046 -0.5971835
## [2,] 0.5971835  0.8021046

# The print method of varimax omits loadings less than abs(0.1). In order to
display all the loadings, it is necessary to ask explicitly the contents of
the object $loadings
fact.load.custc <- rot.fact.custc$loadings[,1:2]
fact.load.custc

##                              PC1         PC2
## custc.tenure         0.9878704 -0.07545929
## custc.MonthlyCharges 0.1624740 -0.98260107
## custc.TotalCharges   0.8194167 -0.54371924

# Computing the rotated factor scores . Notice that signs are reversed for
factors F2 (PC2), F3 (PC3) and F4 (PC4)
scale.custc <- scale(quant_var_df[])
#scale.custc

library(psych)
```

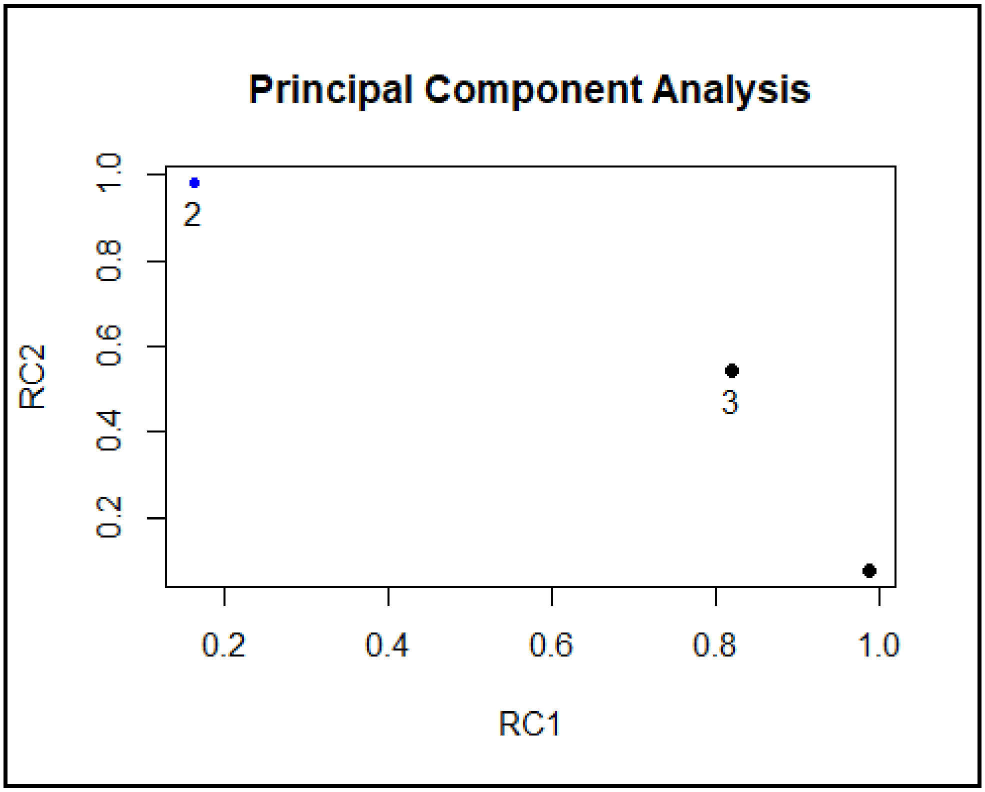
```
## Warning: package 'psych' was built under R version 3.6.2

#install.packages("psych",
lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")
fit.pc <- principal(quant_var_df[], nfactors=2, rotate="varimax")
fit.pc

## Principal Components Analysis
## Call: principal(r = quant_var_df[], nfactors = 2, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                        RC1  RC2   h2     u2 com
## custc.tenure         0.99 0.08 0.98 0.0184 1.0
## custc.MonthlyCharges 0.16 0.98 0.99 0.0081 1.1
## custc.TotalCharges   0.82 0.54 0.97 0.0329 1.7
##
##                         RC1  RC2
## SS loadings           1.67 1.27
## Proportion Var        0.56 0.42
## Cumulative Var        0.56 0.98
## Proportion Explained  0.57 0.43
## Cumulative Proportion 0.57 1.00
##
## Mean item complexity =  1.3
## Test of the hypothesis that 2 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.02
##  with the empirical chi square  14.38  with prob <  NA
##
## Fit based upon off diagonal values = 1

round(fit.pc$values, 4)

## [1] 2.1798 0.7608 0.0594

fit.pc$loadings

##
## Loadings:
##                      RC1   RC2
## custc.tenure         0.988
## custc.MonthlyCharges 0.162 0.983
## custc.TotalCharges   0.819 0.544
##
##                  RC1   RC2
## SS loadings    1.674 1.267
## Proportion Var 0.558 0.422
## Cumulative Var 0.558 0.980

# Loadings with more digits
for (i in c(1,2)) { print(fit.pc$loadings[[1,i]])}
```

```
## [1] 0.9878704
## [1] 0.07545929

# Communalities
fit.pc$communality

##         custc.tenure custc.MonthlyCharges   custc.TotalCharges
##            0.9815821            0.9919027            0.9670743

# Rotated factor scores, Notice the columns ordering: RC1, RC2
fit.pc$scores

##                   RC1           RC2
##    [1,] -1.0035643474 -0.8654501015
##    [2,]  0.0491890247 -0.2994583670
##    [3,] -1.1824292272 -0.1334897906
##    [4,]  0.5005425511 -0.9018579285
##    [5,] -1.3307490150  0.3887333951
##    [6,] -1.3030091091  1.2614036085
##    [7,] -0.5917428972  0.8555073570
##    [8,] -0.6888869310 -0.9635183592
##    [9,] -0.3652107403  1.3609209678
##   [10,]  1.1664581600 -0.5489549830
##   [11,] -0.7367305371 -0.3557300688
##   [12,] -0.4075209170 -1.3736432242
##   [13,]  1.0291780608  1.0697610657
##   [14,]  0.6181128306  1.2327564829
##   [15,] -0.5241714992  1.3888149687
##   [16,]  1.6277837939  1.5270561832
##   [17,]  0.7620193960 -1.7381643354
##   [18,]  1.6562060765  1.2506819813
##   [19,] -0.8843203363 -0.1596617112
##   [20,] -0.6454097185  0.8899893249
##   [21,] -1.0925838807 -0.5631796580
## [7022,] -1.4205027735  0.5502859553
## [7023,]  0.1817426857  0.1019832855
## [7024,]  1.4847875441  1.1423586991
## [7025,] -0.6646712967  0.5332193013
## [7026,] -0.8375982462  0.0013269713
## [7027,]  1.4176052206 -1.9534316627
## [7028,] -0.4856382091  0.7003547915
## [7029,]  1.7146657691  1.1278456155
## [7030,] -0.6500894645 -0.9773404586
## [7031,] -1.2784894792  0.4902741140
## [7032,]  1.4228712906  1.2348562828

# Play with FA utilities
fa.parallel(quant_var_df[]) # See factor recommendation

## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs =
np.obs, :
```

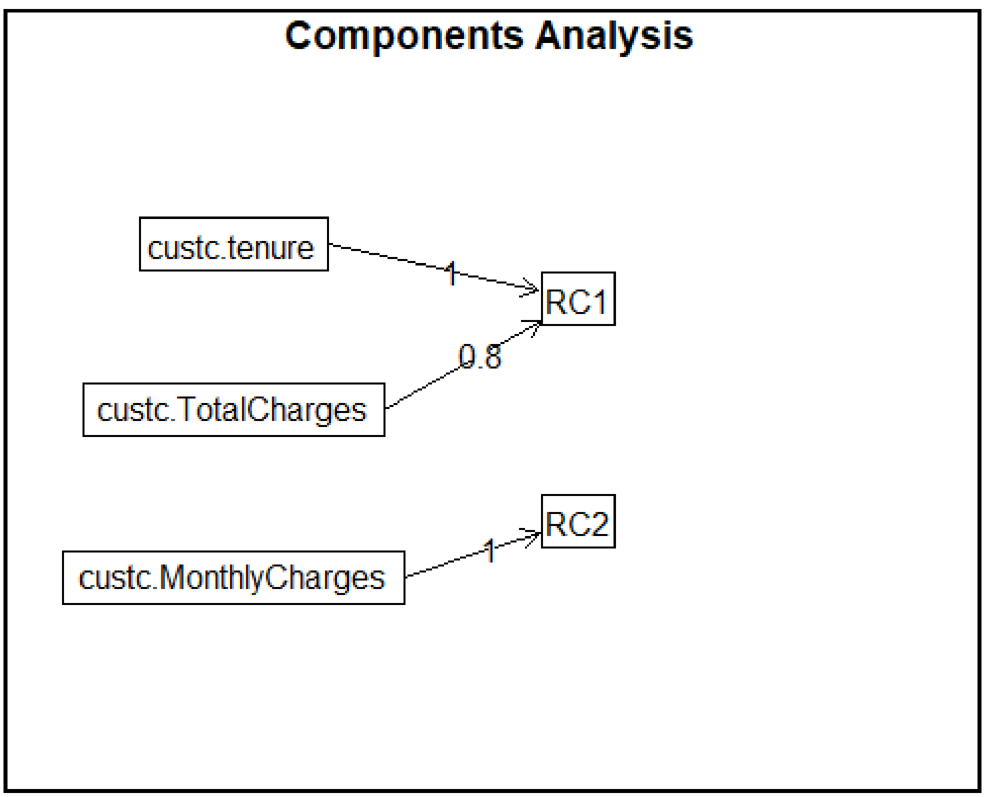
```
## The estimated weights for the factor scores are probably incorrect. Try a
## different factor score estimation method.

## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate,
: An
## ultra-Heywood case was detected. Examine the results carefully
```




```
## Parallel analysis suggests that the number of factors =  2  and the number
of components =  1

fa.plot(fit.pc) # See Correlations within Factors
```

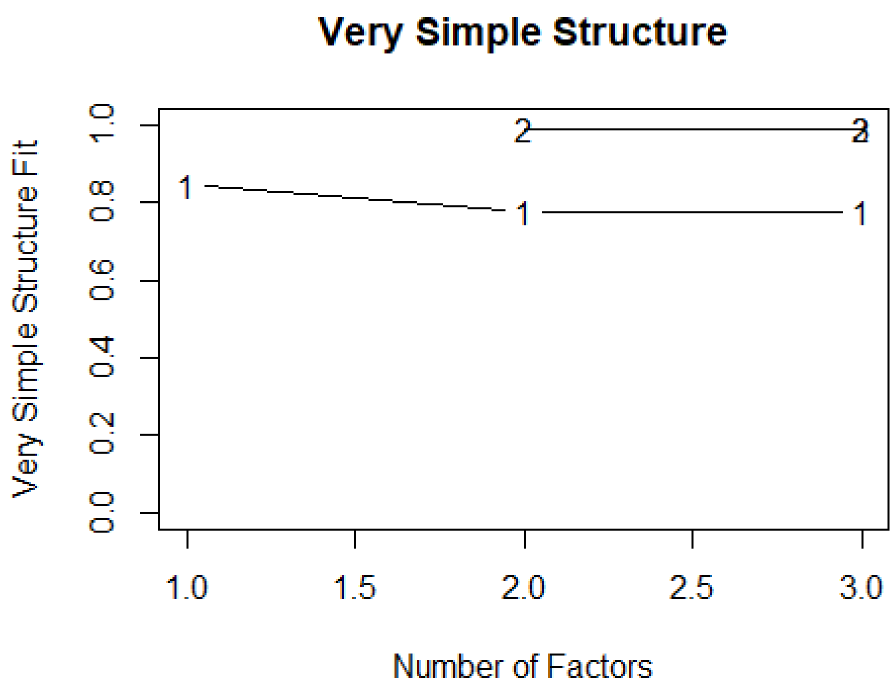



```r
fa.diagram(fit.pc) # Visualize the relationship
```




```r
vss(quant_var_df[]) # See Factor recommendations for a simple structure
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs =
np.obs, :
## The estimated weights for the factor scores are probably incorrect. Try a
## different factor score estimation method.

## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs =
np.obs, :
## An ultra-Heywood case was detected. Examine the results carefully
```




```
##
## Very Simple Structure
## Call: vss(x = quant_var_df[])
## VSS complexity 1 achieves a maximimum of 0.85  with  1  factors
## VSS complexity 2 achieves a maximimum of 0.99  with  2  factors
##
## The Velicer MAP achieves a minimum of NA  with  1  factors
## BIC achieves a minimum of  NA  with    factors
## Sample Size adjusted BIC achieves a minimum of  NA  with    factors
##
## Statistics by number of factors
##   vss1 vss2  map dof   chisq prob sqresid  fit RMSEA BIC SABIC complex
eChisq
## 1 0.85 0.00 0.34   0 4.5e+03   NA   0.816 0.85    NA  NA    NA     1.0
4.6e+02
## 2 0.78 0.99 1.00  -2 3.1e-12   NA   0.058 0.99    NA  NA    NA     1.3
3.0e-14
## 3 0.78 0.99   NA  -3 3.1e-12   NA   0.058 0.99    NA  NA    NA     1.3
```

```
3.0e-14
##       SRMR eCRMS eBIC
## 1 1.0e-01    NA   NA
## 2 8.4e-10    NA   NA
## 3 8.4e-10    NA   NA
```