Istanbul_airbnb_factor_analysis

```r
knitr::opts_chunk$set(echo = TRUE)
```

```r
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.6.2
```

```r
library(fpp)
```

```
## Loading required package: forecast
```

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

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## Loading required package: fma
```

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

```
## Loading required package: expsmooth
```

```
## Loading required package: lmtest
```

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

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

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

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

```
## Loading required package: tseries
```

```r
library(fpp2)
```

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

```
##
## Attaching package: 'fpp2'
```

```
## The following objects are masked from 'package:fpp':
##
##     ausair, ausbeer, austa, austourists, debitcards, departures,
##     elecequip, euretail, guinearice, oil, sunspotarea, usmelec
```

```r
library(cowplot)
```

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

##
## *********************************************************

## Note: As of version 1.0.0, cowplot does not change the

##   default ggplot2 theme anymore. To recover the previous

##   behavior, execute:
##   theme_set(theme_cowplot())

## *********************************************************
```

```r
library(tidyverse)
```

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

## -- Attaching packages -------------------------------------------------
------- tidyverse 1.3.0 --

## v tibble  2.1.3      v dplyr   0.8.4
## v tidyr   1.0.2      v stringr 1.4.0
## v readr   1.3.1      v forcats 0.4.0
## v purrr   0.3.3

## Warning: package 'tidyr' was built under R version 3.6.2

## Warning: package 'purrr' was built under R version 3.6.2

## Warning: package 'dplyr' was built under R version 3.6.2

## Warning: package 'forcats' was built under R version 3.6.2

## -- Conflicts ----------------------------------------------------------
- tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x purrr::transpose() masks data.table::transpose()
```

```r
library(psych)
```

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

##
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```r
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.6.2
```

```r
library(dplyr)
library(corrplot)
```

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

```
## corrplot 0.84 loaded
```

```r
library(GGally)
```

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

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```
## The following object is masked from 'package:fma':
##
##     pigs
```

```r
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

```r
AirbnbIstanbul <- read.csv("C:/Pritesh/Rutgers/Courses/Projects/MVA/Dataset/A
irbnbIstanbul.csv", stringsAsFactors=FALSE)
Istanbul <- copy(AirbnbIstanbul)
class(Istanbul)
```

```
## [1] "data.frame"
```

```r
setDT(Istanbul)
```

```r
str(Istanbul)
```

```
## Classes 'data.table' and 'data.frame':   16251 obs. of  16 variables:
##  $ id                            : int  4826 20815 25436 27271 28277 28308
28318 29241 30697 33368 ...
##  $ name                          : chr  "The Place" "The Bosphorus from Th
e Comfy Hill" "House for vacation rental furnutare" "LOVELY APT. IN PERFECT L
OCATION" ...
##  $ host_id                       : int  6603 78838 105823 117026 121607 12
1695 121721 125742 132137 135136 ...
##  $ host_name                     : chr  "Kaan" "GÃ¼lder" "Yesim" "Mutlu" .
..
##  $ neighbourhood_group           : logi  NA NA NA NA NA NA ...
##  $ neighbourhood                 : chr  "Uskudar" "Besiktas" "Besiktas" "B
eyoglu" ...
##  $ latitude                      : num  41.1 41.1 41.1 41 41 ...
##  $ longitude                     : num  29.1 29 29 29 29 ...
##  $ room_type                     : chr  "Entire home/apt" "Entire home/apt
" "Entire home/apt" "Entire home/apt" ...
##  $ price                         : int  554 100 211 237 591 237 633 264 59
6 295 ...
##  $ minimum_nights                : int  1 30 21 5 3 1 3 3 1 2 ...
##  $ number_of_reviews             : int  1 41 0 2 0 0 0 0 1 1 ...
##  $ last_review                   : chr  "2009-06-01" "2018-11-07" "" "2018
-05-04" ...
##  $ reviews_per_month             : num  0.01 0.38 NA 0.04 NA NA NA NA 0.01
0.02 ...
##  $ calculated_host_listings_count: int  1 2 1 1 13 1 1 1 1 2 ...
##  $ availability_365              : int  365 49 83 228 356 365 365 365 365
232 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

Factoring categorical variables

```r
Istanbul[,room_type:=factor(room_type)]
Istanbul[,neighbourhood:=factor(neighbourhood)]
Istanbul[,last_review:=as.Date(last_review,'%Y-%m-%d')] ## converting last_re
view to date datatype

# datatypes looks better now. hence will see again for NA values
grep ('NA',Istanbul) # 2, 5, 13 and 14 column have NA values
```

```
## [1]  2  5 13 14
```

```r
Istanbul[is.na(neighbourhood_group),NROW(neighbourhood_group)] # entire obs.
is blank, will drop this var
```

```
## [1] 16251
```

```r
Istanbul[is.na(last_review),NROW(last_review)] ## there are 8484 NA values
```

```
## [1] 8484
```

```r
Istanbul[is.na(reviews_per_month),NROW(reviews_per_month)] ## there are 8484
NA values
```

```
## [1] 8484
```

```r
Istanbul$neighbourhood_group <- NULL ## removing neighbourhood_group column
Istanbul[is.na(reviews_per_month),reviews_per_month:=0] ## nearly 50% of the
dataset is filled with NA.
# hence we can't simply remove these many rows. Hence imputing with 0 values.

nrow(Istanbul[price > 1000]) ## price > 1000
```

```
## [1] 613
```

```r
#Only 613 rows out of 16251 have Price>1000 which are outliers as seen in EDA
, we can remove those records

Istanbul <- Istanbul[price < 1000] # removing outliers [1] 15638    15
dim(Istanbul)
```

```
## [1] 15638    15
```

Creating new data table with all the quantitative column named Istanbul_factor

```r
Istanbul_factor <- Istanbul[,c("latitude","longitude","price","minimum_nights
","number_of_reviews","reviews_per_month","calculated_host_listings_count","a
vailability_365")]
corrm.Istanbul <- cor(Istanbul_factor)
corrm.Istanbul
```

```
##                                    latitude    longitude        price
## latitude                         1.000000000 -0.197168094  0.054487580
## longitude                       -0.197168094  1.000000000 -0.035045643
## price                            0.054487580 -0.035045643  1.000000000
## minimum_nights                   0.001806824 -0.008447202  0.003237415
## number_of_reviews               -0.020171577 -0.002091917  0.020700048
## reviews_per_month               -0.030645872  0.009699078 -0.025490933
## calculated_host_listings_count  -0.009835884 -0.034267106  0.079463904
## availability_365                -0.005504686 -0.038766412  0.160241947
##                                 minimum_nights number_of_reviews
## latitude                          0.001806824       -0.020171577
## longitude                        -0.008447202       -0.002091917
## price                             0.003237415        0.020700048
## minimum_nights                    1.000000000       -0.013837757
## number_of_reviews                -0.013837757        1.000000000
## reviews_per_month                -0.034105874        0.576543022
## calculated_host_listings_count   -0.017881502        0.181090297
## availability_365                  0.012869263        0.048541558
##                                 reviews_per_month calculated_host_listings_
count
```

```
## latitude                                -0.030645872                -0.0098
35884
## longitude                                 0.009699078                -0.0342
67106
## price                                    -0.025490933                 0.0794
63904
## minimum_nights                           -0.034105874                -0.0178
81502
## number_of_reviews                         0.576543022                 0.1810
90297
## reviews_per_month                         1.000000000                 0.1081
87924
## calculated_host_listings_count            0.108187924                 1.0000
00000
## availability_365                         -0.007430996                 0.1677
18740
##                                      availability_365
## latitude                               -0.005504686
## longitude                              -0.038766412
## price                                   0.160241947
## minimum_nights                          0.012869263
## number_of_reviews                       0.048541558
## reviews_per_month                      -0.007430996
## calculated_host_listings_count          0.167718740
## availability_365                        1.000000000

plot(corrm.Istanbul)
```
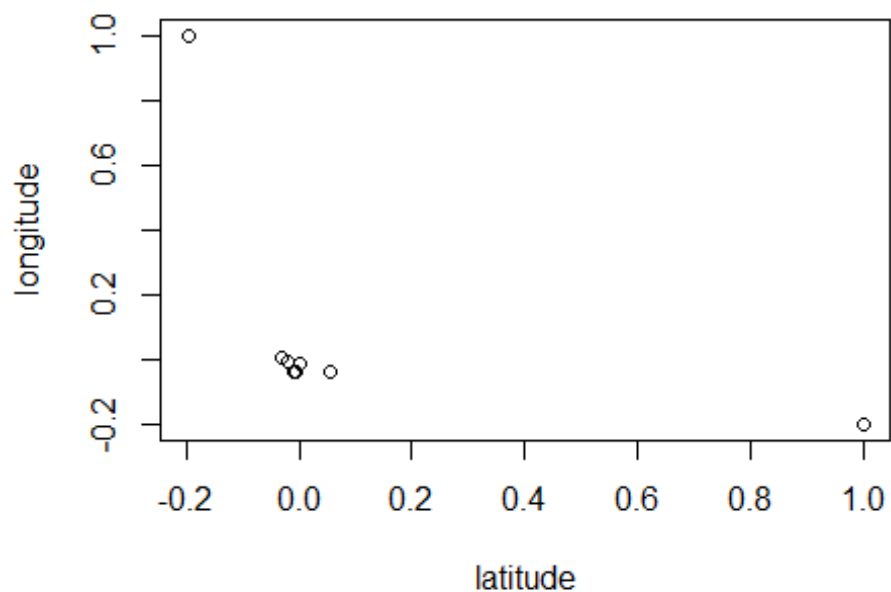
```
Istanbul_pca <- prcomp(Istanbul_factor, scale=TRUE)
summary(Istanbul_pca)

## Importance of components:
##                            PC1    PC2    PC3    PC4    PC5    PC6     PC7
## Standard deviation      1.2880 1.1367 1.0725 0.9995 0.9482 0.8963 0.88367
## Proportion of Variance 0.2074 0.1615 0.1438 0.1249 0.1124 0.1004 0.09761
## Cumulative Proportion  0.2074 0.3689 0.5127 0.6375 0.7499 0.8503 0.94795
##                            PC8
## Standard deviation     0.64529
## Proportion of Variance 0.05205
## Cumulative Proportion  1.00000

plot(Istanbul_pca)
```
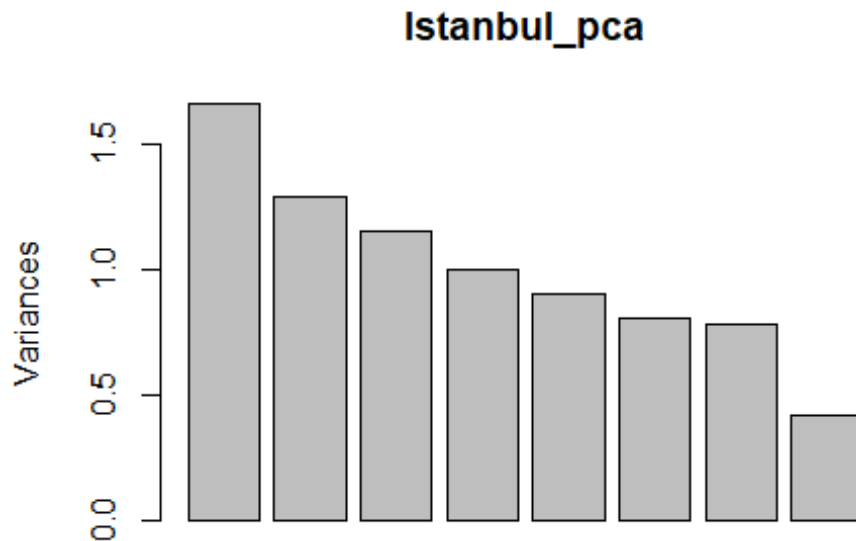

Istanbul_pca

```
# A table containing eigenvalues and %'s accounted, follows. Eigenvalues are
the sdev^2
(eigen_Istanbul <- round(Istanbul_pca$sdev^2,2))

## [1] 1.66 1.29 1.15 1.00 0.90 0.80 0.78 0.42

names(eigen_Istanbul) <- paste("PC",1:8,sep="")
eigen_Istanbul

##  PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8
## 1.66 1.29 1.15 1.00 0.90 0.80 0.78 0.42
```
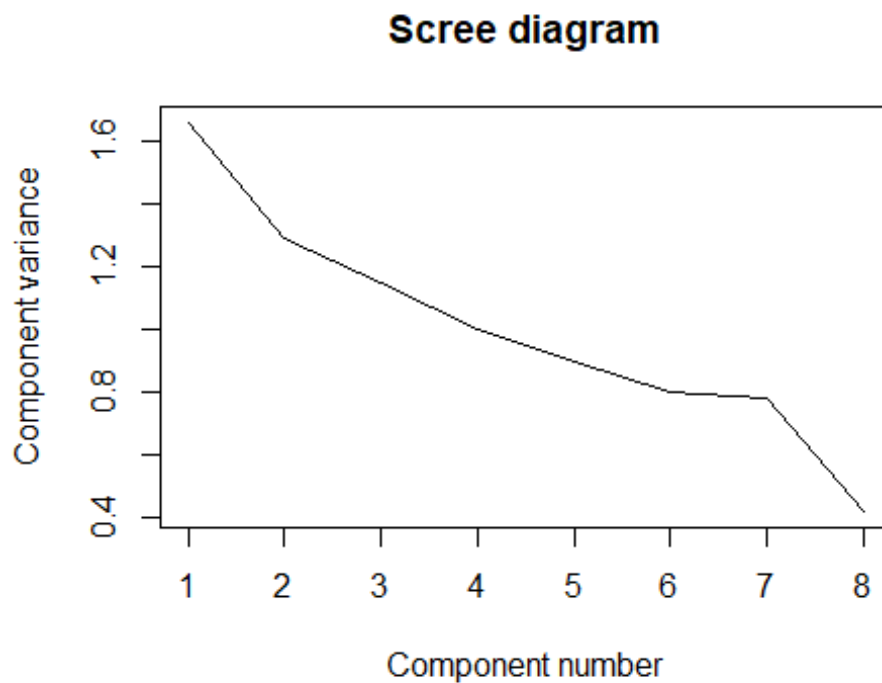
```
plot(eigen_Istanbul, xlab = "Component number", ylab = "Component variance",
type = "l", main = "Scree diagram")
```
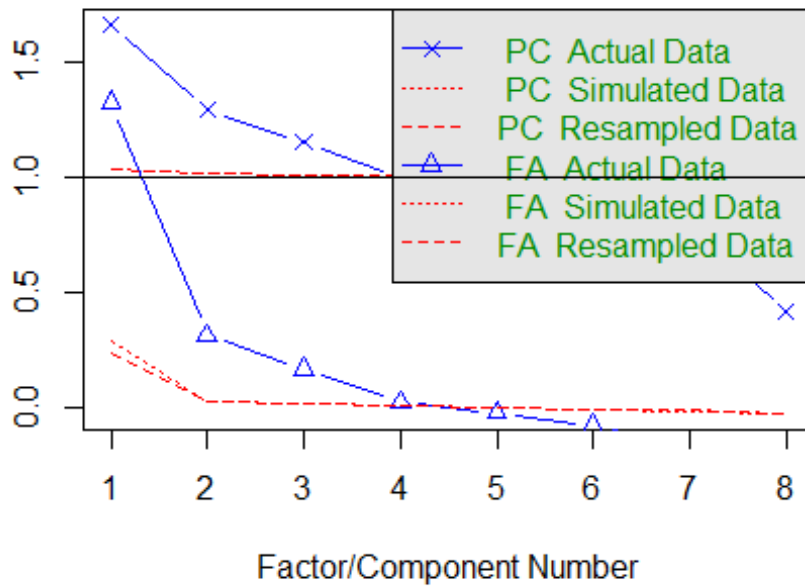
Scree diagram

As per scree plot, there should be 7 factors, will see what parallel analysis sugge sts

```
sumlambdas <- sum(eigen_Istanbul) ## eigen values
sumlambdas
```

```
## [1] 8
```

```
fa.parallel(Istanbul_factor)
```

Parallel Analysis Scree Plots

eigenvalues of principal components and factor analys

Legend:
- PC Actual Data
- PC Simulated Data
- PC Resampled Data
- FA Actual Data
- FA Simulated Data
- FA Resampled Data

Factor/Component Number
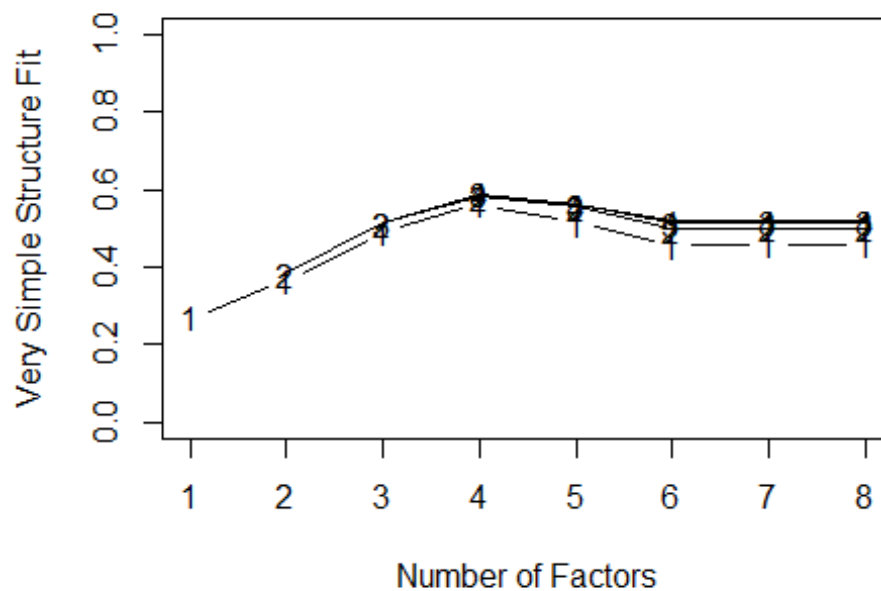
Parallel analysis suggests that the number of factors = 4 and the number of components = 3

```
vss(Istanbul_factor) # See Factor recommendations for a simple structure
```

Very Simple Structure


```
## 
## Very Simple Structure
## Call: vss(x = Istanbul_factor)
## VSS complexity 1 achieves a maximimum of 0.56  with  4  factors
## VSS complexity 2 achieves a maximimum of 0.58  with  4  factors
## 
## The Velicer MAP achieves a minimum of NA  with  1  factors
## BIC achieves a minimum of  NA  with  3  factors
## Sample Size adjusted BIC achieves a minimum of  NA  with  3  factors
## 
## Statistics by number of factors
##    vss1 vss2   map dof   chisq     prob sqresid  fit  RMSEA  BIC SABIC comp
lex
## 1 0.27 0.00 0.032  20 1.7e+03  0.0e+00     6.5 0.27 0.0726 1477  1540
1.0
## 2 0.36 0.38 0.056  13 6.8e+02 6.1e-137     5.5 0.38 0.0573  555   596
1.2
## 3 0.49 0.51 0.095   7 2.9e+01  1.3e-04     4.4 0.51 0.0142  -38   -16
1.1
## 4 0.56 0.58 0.153   2 2.6e+00  2.8e-01     3.7 0.59 0.0043  -17   -10
1.3
## 5 0.51 0.56 0.259  -2 1.2e-06       NA     3.9 0.56     NA   NA    NA
1.2
## 6 0.46 0.50 0.532  -5 1.2e-07       NA     4.3 0.52     NA   NA    NA
1.3
## 7 0.46 0.50 1.000  -7 3.6e-09       NA     4.3 0.52     NA   NA    NA
1.3
```

```
## 8 0.46 0.50     NA   -8 3.6e-09         NA      4.3 0.52     NA    NA    NA
1.3
##     eChisq    SRMR  eCRMS eBIC
## 1 3.3e+03 6.2e-02 0.0728 3120
## 2 1.3e+03 3.8e-02 0.0555 1125
## 3 3.1e+01 5.9e-03 0.0118  -37
## 4 3.3e+00 1.9e-03 0.0072  -16
## 5 1.1e-06 1.1e-06     NA   NA
## 6 1.4e-07 4.0e-07     NA   NA
## 7 3.2e-09 6.0e-08     NA   NA
## 8 3.2e-09 6.0e-08     NA   NA

# VSS complexity 1 achieves a maximimum of 0.56  with  4  factors
# VSS complexity 2 achieves a maximimum of 0.58  with  4  factors
#
# The Velicer MAP achieves a minimum of NA  with  1  factors
# BIC achieves a minimum of  NA  with  3  factors
# Sample Size adjusted BIC achieves a minimum of  NA  with  3  factors


nfactors(Istanbul_factor)
```
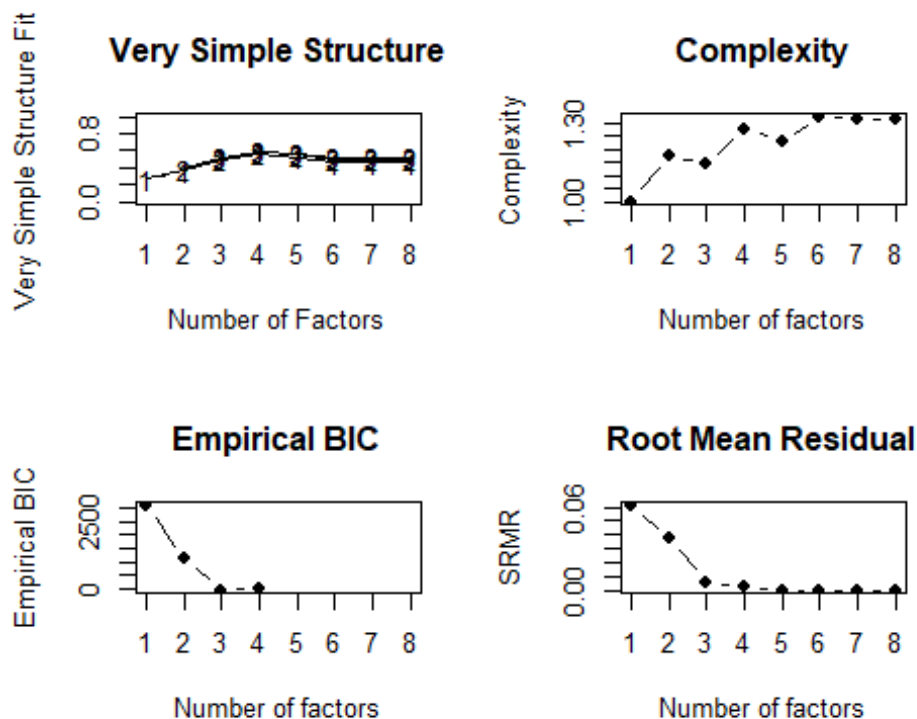


```
##
## Number of factors
## Call: vss(x = x, n = n, rotate = rotate, diagonal = diagonal, fm = fm,
##     n.obs = n.obs, plot = FALSE, title = title, use = use, cor = cor)
## VSS complexity 1 achieves a maximimum of 0.56  with  4  factors
## VSS complexity 2 achieves a maximimum of 0.58  with  4  factors
```

```
## The Velicer MAP achieves a minimum of 0.03  with  1  factors
## Empirical BIC achieves a minimum of  -36.95  with  3  factors
## Sample Size adjusted BIC achieves a minimum of  -16.19  with  3  factors
##
## Statistics by number of factors
##   vss1 vss2  map dof  chisq      prob sqresid  fit  RMSEA  BIC SABIC comp
lex
## 1 0.27 0.00 0.032  20 1.7e+03  0.0e+00     6.5 0.27 0.0726 1477  1540
1.0
## 2 0.36 0.38 0.056  13 6.8e+02 6.1e-137     5.5 0.38 0.0573  555   596
1.2
## 3 0.49 0.51 0.095   7 2.9e+01  1.3e-04     4.4 0.51 0.0142  -38   -16
1.1
## 4 0.56 0.58 0.153   2 2.6e+00  2.8e-01     3.7 0.59 0.0043  -17   -10
1.3
## 5 0.51 0.56 0.259  -2 1.2e-06       NA     3.9 0.56     NA   NA    NA
1.2
## 6 0.46 0.50 0.532  -5 1.2e-07       NA     4.3 0.52     NA   NA    NA
1.3
## 7 0.46 0.50 1.000  -7 3.6e-09       NA     4.3 0.52     NA   NA    NA
1.3
## 8 0.46 0.50    NA  -8 3.6e-09       NA     4.3 0.52     NA   NA    NA
1.3
##    eChisq    SRMR  eCRMS eBIC
## 1 3.3e+03 6.2e-02 0.0728 3120
## 2 1.3e+03 3.8e-02 0.0555 1125
## 3 3.1e+01 5.9e-03 0.0118  -37
## 4 3.3e+00 1.9e-03 0.0072  -16
## 5 1.1e-06 1.1e-06     NA   NA
## 6 1.4e-07 4.0e-07     NA   NA
## 7 3.2e-09 6.0e-08     NA   NA
## 8 3.2e-09 6.0e-08     NA   NA
```

nfactors suggests we can either go with 3 factors or 4 factors

```
# Part 1, with four factors
library(psych)
fit.pc4 <- principal(Istanbul_factor, nfactors=4, rotate="varimax")
fit.pc4 #4 factors RC1, RC2, RC3, RC4 are created

## Principal Components Analysis
## Call: principal(r = Istanbul_factor, nfactors = 4, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                             RC1   RC2   RC3   RC4   h2     u2 com
## latitude                  -0.03 -0.02  0.78 -0.02 0.61 0.3877 1.0
## longitude                 -0.02 -0.07 -0.76 -0.02 0.58 0.4181 1.0
## price                     -0.09  0.62  0.11  0.00 0.41 0.5926 1.1
## minimum_nights            -0.01  0.00  0.00  1.00 0.99 0.0055 1.0
## number_of_reviews          0.88  0.09  0.00  0.01 0.77 0.2263 1.0
## reviews_per_month          0.87 -0.04 -0.01 -0.02 0.77 0.2320 1.0
```

```
## calculated_host_listings_count  0.28   0.57 -0.02 -0.06 0.40 0.5989 1.5
## availability_365                -0.03   0.75 -0.04  0.05 0.56 0.4387 1.0
##
##                              RC1  RC2  RC3  RC4
## SS loadings            1.62 1.28 1.20 1.00
## Proportion Var         0.20 0.16 0.15 0.13
## Cumulative Var         0.20 0.36 0.51 0.64
## Proportion Explained  0.32 0.25 0.24 0.20
## Cumulative Proportion 0.32 0.57 0.80 1.00
##
## Mean item complexity =  1.1
## Test of the hypothesis that 4 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.13
##  with the empirical chi square  14145.13  with prob <  0
##
## Fit based upon off diagonal values = 0.08
```

```r
round(fit.pc4$values, 3)
```

```
## [1] 1.659 1.292 1.150 0.999 0.899 0.803 0.781 0.416
```

```r
#Above are factor values for all 8 variables
fit.pc4$loadings
```

```
##
## Loadings:
##                                 RC1     RC2     RC3     RC4
## latitude                                        0.781
## longitude                                      -0.759
## price                                   0.623   0.105
## minimum_nights                                          0.997
## number_of_reviews            0.875
## reviews_per_month            0.875
## calculated_host_listings_count  0.278   0.565
## availability_365                        0.746
##
##                  RC1   RC2   RC3   RC4
## SS loadings    1.620 1.278 1.200 1.002
## Proportion Var 0.202 0.160 0.150 0.125
## Cumulative Var 0.202 0.362 0.512 0.638
```

```r
# Above are the Loadings for all 8 variables

for (i in c(1,2,3,4)) { print(fit.pc4$loadings[[1,i]])}
```

```
## [1] -0.03077102
## [1] -0.02069587
## [1] 0.7814196
## [1] -0.01709352
```

Communalities

```
fit.pc4$communality
```

```
##                       latitude                        longitude
##                      0.6122840                        0.5818972
##                          price                   minimum_nights
##                      0.4074236                        0.9945396
##              number_of_reviews                reviews_per_month
##                      0.7737051                        0.7679744
## calculated_host_listings_count                  availability_365
##                      0.4010889                        0.5612598
```

```
#Above are the communalities for all 8 variabbles
```

Rotated factor scores

```
head(fit.pc4$scores)
```

```
##               RC1         RC2        RC3         RC4
## [1,] -0.7188928  1.2150301 0.14835545 -0.11114975
## [2,]  0.9190461 -1.3792418 0.41991784  0.85933486
## [3,] -0.4308853 -0.9829675 0.56648630  0.49807054
## [4,] -0.4308808 -0.2203644 0.13281517  0.02074727
## [5,] -0.5315036  1.9813454 0.30817477 -0.12110005
## [6,] -0.5697539  0.3782424 0.04495653 -0.07605584
```

```
fa.diagram(fit.pc4) # To Visualize the relationship and mapping between varia
bles and factors with weights
```

Components Analysis
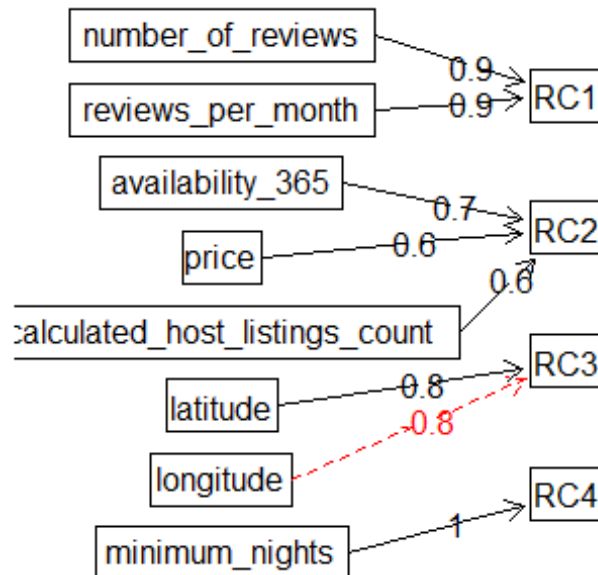
Above, output gives weigths going in RCs
red line indicates negative relation

As per above diagram, all the factors have significant contribution and so
its better not to loose any of 4 factors
So we will take all four RC1, RC2, RC3 and RC4 as inputs for our models

Above factor analysis, we can conclude to reduce number of variables from 8 to 4 in our input dataset.

```
#Now lets rename these factors as per their contributing variables
colnames(fit.pc4$loadings) <- c("NumReviews_reviewspm","Prie_Availability_Hos
tListCount","minNights","GeoLocations")
fit.pc4$loadings

##
## Loadings:
##                                NumReviews_reviewspm
## latitude
## longitude
## price
## minimum_nights
## number_of_reviews              0.875
## reviews_per_month              0.875
## calculated_host_listings_count 0.278
```

```
## availability_365
##                                    Prie_Availability_HostListCount minNights
## latitude                                                            0.781
## longitude                                                          -0.759
## price                                0.623                          0.105
## minimum_nights
## number_of_reviews
## reviews_per_month
## calculated_host_listings_count      0.565
## availability_365                    0.746
##                                    GeoLocations
## latitude
## longitude
## price
## minimum_nights                       0.997
## number_of_reviews
## reviews_per_month
## calculated_host_listings_count
## availability_365
##
##               NumReviews_reviewspm Prie_Availability_HostListCount minNig
hts
## SS loadings              1.620                                    1.278     1.
200
## Proportion Var           0.202                                    0.160     0.
150
## Cumulative Var           0.202                                    0.362     0.
512
##               GeoLocations
## SS loadings        1.002
## Proportion Var     0.125
## Cumulative Var     0.638

#Plotting the correlation beyween these factors
plot(fit.pc4)
```

Principal Component Analysis
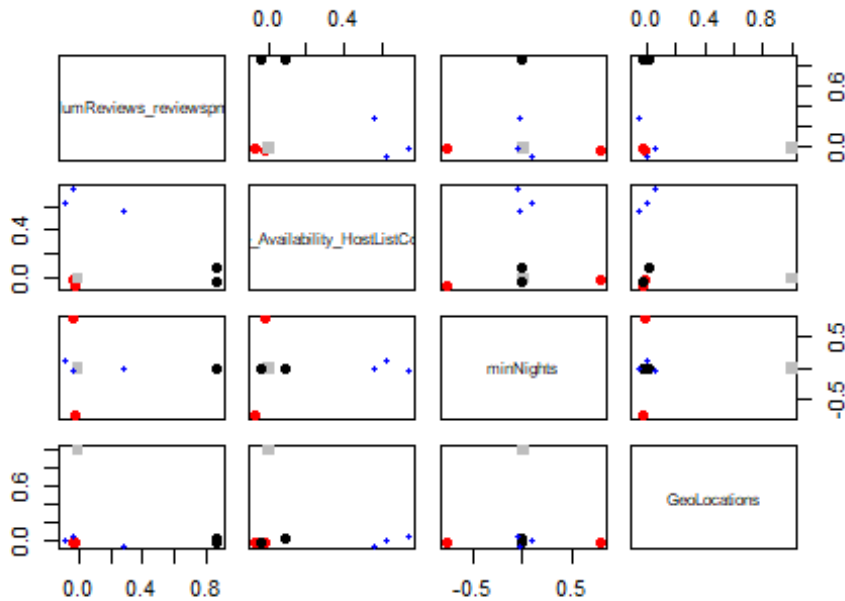

```r
# Part 2, with three factors
library(psych)
fit.pc3 <- principal(Istanbul_factor, nfactors=3, rotate="varimax")
fit.pc3 #3 factors RC1, RC2, RC3 are created

## Principal Components Analysis
## Call: principal(r = Istanbul_factor, nfactors = 3, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                                RC1   RC2   RC3    h2    u2 com
## latitude                     -0.04 -0.03  0.78 0.6115 0.39 1.0
## longitude                     0.00 -0.07 -0.76 0.5818 0.42 1.0
## price                        -0.14  0.61  0.10 0.4063 0.59 1.2
## minimum_nights               -0.09  0.03  0.01 0.0087 0.99 1.4
## number_of_reviews             0.86  0.15  0.02 0.7690 0.23 1.1
## reviews_per_month             0.87  0.02  0.01 0.7661 0.23 1.0
## calculated_host_listings_count 0.24 0.58 -0.02 0.3969 0.60 1.3
## availability_365             -0.09  0.74 -0.04 0.5610 0.44 1.0
##
##                       RC1  RC2  RC3
## SS loadings          1.60 1.30 1.20
## Proportion Var       0.20 0.16 0.15
## Cumulative Var       0.20 0.36 0.51
## Proportion Explained 0.39 0.32 0.29
## Cumulative Proportion 0.39 0.71 1.00
##
## Mean item complexity =  1.1
## Test of the hypothesis that 3 components are sufficient.
```

```
##
## The root mean square of the residuals (RMSR) is  0.13
##  with the empirical chi square  14219.67  with prob <  0
##
## Fit based upon off diagonal values = 0.07

round(fit.pc3$values, 3)

## [1] 1.659 1.292 1.150 0.999 0.899 0.803 0.781 0.416
```

#Above are factor values for all 8 variables

```
fit.pc3$loadings

##
## Loadings:
##                                  RC1    RC2    RC3
## latitude                                    0.780
## longitude                                  -0.760
## price                          -0.140  0.613  0.104
## minimum_nights
## number_of_reviews               0.863  0.154
## reviews_per_month               0.875
## calculated_host_listings_count  0.240  0.582
## availability_365                       0.743
##
##                  RC1   RC2   RC3
## SS loadings    1.605 1.297 1.199
## Proportion Var 0.201 0.162 0.150
## Cumulative Var 0.201 0.363 0.513
```

Above are the Loadings for all 8 variables

```
for (i in c(1,2,3)) { print(fit.pc3$loadings[[1,i]])}

## [1] -0.04497776
## [1] -0.02607995
## [1] 0.7802823
```

Communalities
```
fit.pc3$communality

##                        latitude                      longitude
##                     0.611543631                    0.581767090
##                           price                  minimum_nights
##                     0.406251294                    0.008702142
##               number_of_reviews               reviews_per_month
##                     0.769040100                    0.766133759
## calculated_host_listings_count                availability_365
##                     0.396866242                    0.560965370
```
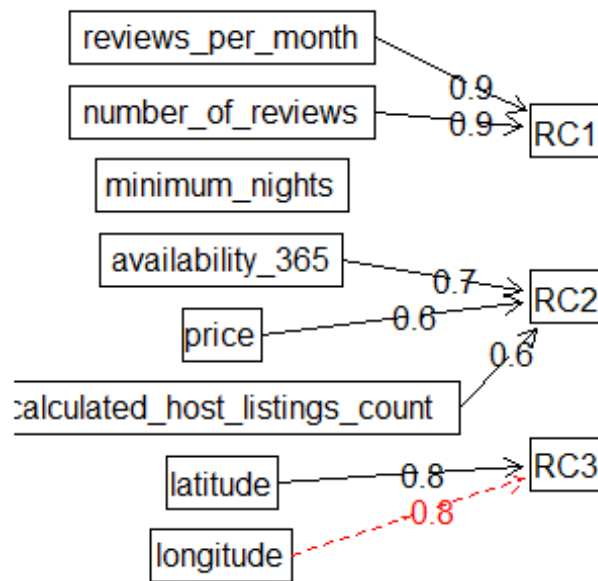
fa.diagram(fit.pc3) *# To Visualize the relationship and mapping between varia bles and factors with weights*

Components Analysis

reviews_per_month

number_of_reviews

minimum_nights

availability_365

price

calculated_host_listings_count

latitude

longitude

RC1 0.9 0.9

RC2 0.7 0.6 0.6

RC3 0.8 -0.8

Above, output gives weigths going in RCs
red line indicates negative relation

As per above diagram, all the factors have significant contribution and so
its better not to loose any of 3 factors
So we will take all four RC1, RC2 and RC3 as inputs for our models
 We can see that minimum_nights doesn't have any contribution, hence we can consider dropping this variable

From Above factor analysis, we can conclude to reduce number of variables from 8 to 3 in our input dataset.

#Now lets rename these factors as per their contributing variables
colnames(fit.pc3$loadings) <- **c**("NumReviews_reviewspm","Prie_Avail_HostListCount","GeoLocations")
fit.pc3$loadings

##
Loadings:

```
##                              NumReviews_reviewspm Prie_Avail_HostListCou
nt
## latitude
## longitude
## price                                      -0.140                   0.613
## minimum_nights
## number_of_reviews                           0.863                   0.154
## reviews_per_month                           0.875
## calculated_host_listings_count   0.240                   0.582
## availability_365                                                    0.743
##                              GeoLocations
## latitude                        0.780
## longitude                      -0.760
## price                           0.104
## minimum_nights
## number_of_reviews
## reviews_per_month
## calculated_host_listings_count
## availability_365
##
##                 NumReviews_reviewspm Prie_Avail_HostListCount GeoLocations
## SS loadings                    1.605                    1.297        1.199
## Proportion Var                 0.201                    0.162        0.150
## Cumulative Var                 0.201                    0.363        0.513
```

```
#Plotting the correlation beyween these factors
plot(fit.pc3)
```


Principal Component Analysis
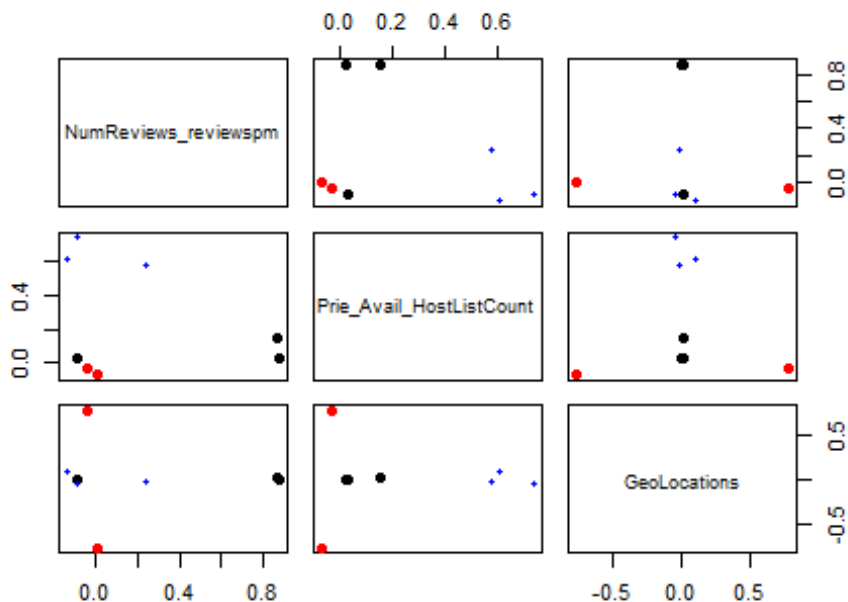
- If we use only 3 variables then we are losing variance from the column : 'minimum_nights' which will cause loss of information.

- Thus, we use factor analysis with 4 factors: RC1, RC2, RC3 and RC4 as inputs for our model.

- As per this factor analysis, we can have reduced number of variables from 8 to 4 in our input dataset.