final project Yuefei Chen

Yuefei Chen

2024-04-29

final project

Question 1. Explain the data collection process. (10 points)

```r
data <- read.csv(file = 'Dataset/Rent_House.csv', header = TRUE, sep = ',')
names(data)
```

```
##  [1] "area"           "rooms"          "bathroom"       "parking.spaces"
##  [5] "floor"          "animal"         "furniture"      "hoa"
##  [9] "rent.amount"    "property.tax"   "fire.insurance"
```

ANS:

In this dataset, The dependent variable is "rent.amount", and independent variables are "area", "rooms", "bathroom", "parking.spaces", "floor", "animal", "furniture", "hoa", property.tax", "fire.insurance" In this dataset,

The "area" is the house area.

The "rooms" represents quantity of rooms.

The "bathrooms" means quantity of bathroom.

The "floor" is the floor of each house. It is a character because some of elements are '-' if the elements is unknown.

The "animal" means whether accept animals or not. It is a boolean variable.

The "parking.spaces" is quantity of parking spaces.

The "hoa" is homeowners association tax.

The "fire.insurance" is fire insurance.

The "property.tax" is property tax.

The "furniture" is with furniture or not.

The "rent.amount" is rent price.

The range of data are as follows.

```r
summary(data)
```

```
##       area            rooms           bathroom      parking.spaces
##  Min.   :  11.0   Min.   : 1.000   Min.   : 1.000   Min.   : 0.000
##  1st Qu.:  56.0   1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.: 0.000
##  Median :  90.0   Median : 2.000   Median : 2.000   Median : 1.000
##  Mean   : 141.2   Mean   : 2.505   Mean   : 2.234   Mean   : 1.607
##  3rd Qu.: 182.0   3rd Qu.: 3.000   3rd Qu.: 3.000   3rd Qu.: 2.000
##  Max.   :2000.0   Max.   :13.000   Max.   :10.000   Max.   :12.000
##     floor              animal            furniture              hoa
##  Length:10677       Length:10677       Length:10677       Min.   :   0.0
##  Class :character   Class :character   Class :character   1st Qu.: 170.0
##  Mode  :character   Mode  :character   Mode  :character   Median : 557.0
##                                                           Mean   : 910.2
##                                                           3rd Qu.:1229.0
##                                                           Max.   :9900.0
##   rent.amount     property.tax     fire.insurance
##  Min.   :  450   Min.   :    0.0   Min.   :  3.00
##  1st Qu.: 1529   1st Qu.:   38.0   1st Qu.: 21.00
##  Median : 2650   Median :  125.0   Median : 36.00
##  Mean   : 3891   Mean   :  335.5   Mean   : 53.23
##  3rd Qu.: 5000   3rd Qu.:  375.0   3rd Qu.: 68.00
##  Max.   :45000   Max.   :28120.0   Max.   :677.00
```

Question 2. Exploratory Data Analysis and Visualizations (50 points)

```r
library(MASS)
library(ggplot2)
library(memisc)
```

```
## Loading required package: lattice

##
## Attaching package: 'memisc'

## The following object is masked from 'package:ggplot2':
##
##     syms

## The following objects are masked from 'package:stats':
##
##     contr.sum, contr.treatment, contrasts

## The following object is masked from 'package:base':
##
##     as.array
```

```r
library(ROCR)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:memisc':
##
##     collect, recode, rename, syms

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(klaR)
library(NbClust)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(psych)
```

```
##
## Attaching package: 'psych'

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

```
library(readr)
house_data <- read_csv("Dataset/Rent_House_random_200_multi_regression.csv")
```

```
## Rows: 200 Columns: 11

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): floor, animal, furniture
## dbl (8): area, rooms, bathroom, parking_spaces, hoa, rent_amount, property_t...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
house_data <- house_data[, c(1:4, 6:11)]
house_data <- house_data[,-c(5)]
str(house_data)
```

```
## tibble [200 x 9] (S3: tbl_df/tbl/data.frame)
##  $ area          : num [1:200] 120 45 50 35 204 177 15 70 180 180 ...
##  $ rooms         : num [1:200] 3 1 2 1 4 3 1 2 3 4 ...
##  $ bathroom      : num [1:200] 4 1 1 1 4 3 1 2 3 4 ...
##  $ parking_spaces: num [1:200] 3 1 1 0 2 4 0 1 2 2 ...
##  $ furniture     : chr [1:200] "not furnished" "furnished" "not furnished" "not furnished" ...
##  $ hoa           : num [1:200] 1350 3000 226 260 0 2700 0 1800 700 2600 ...
##  $ rent_amount   : num [1:200] 5600 5520 750 1400 3440 6900 1200 4200 2700 2000 ...
##  $ property_tax  : num [1:200] 560 0 0 0 100 509 0 250 175 584 ...
##  $ fire_insurance: num [1:200] 71 70 10 18 62 89 16 55 40 26 ...
```

The Mahalanobis distance is used to compute the distance between the countries across the different dimensions. The output means the distance between the mean value and each data point.

```r
house_x <- house_data[, c(1:4,6,7:9)]
house_cm <- colMeans(house_x)
house_S <- cov(house_x)
house_MD <- mahalanobis(house_x, house_cm, house_S)
house_MD
```

```
##   [1]   6.7032251  11.1256015   1.5416174   1.6032966   8.4327649   8.0649279
##   [7]   1.8151570   2.2105298   1.8726203  11.3249854   1.0731392   3.1712205
##  [13]   2.6387283   1.6524809   1.3954223   4.0850726  85.2736574   2.2306188
##  [19]   2.2247258   1.7025025   1.6711798   3.3003993   9.5971126  49.9766518
##  [25]   5.7379311   2.3890956   1.8214012   1.9240636   0.8521328  10.8894562
##  [31]   2.4028838   6.0889456   2.1812407   1.5282618  10.9843333  48.6505452
##  [37]  61.1535403   2.6368937   1.6806407   5.5309154   1.9474030   6.3265671
##  [43]   1.4307333   3.9473158   4.9975995  21.1454073  18.6002237   4.0932425
##  [49]  27.2506047   1.8216918   1.8232359   0.8611666   2.1841873   9.6110394
##  [55] 142.3007965   1.6379741   1.5910413   1.9841982   1.8404570   1.5069879
##  [61]   1.9421292   3.6522957   7.8350675   2.4374767  37.5911140   1.7651447
##  [67]   2.4471197   1.7708850   1.7395325   1.9112336   1.7266574   8.5658801
##  [73]   8.9338566   1.7794760   2.6516565   3.1401084   1.5818995   1.7856355
##  [79]   2.4138232   5.4973723   4.0284683   1.2598866   6.1635429  16.4349594
##  [85]   3.0735613   1.4133372  13.1454651   1.7260393   8.9323527   4.0463488
##  [91]   9.8237522   6.6319068  47.5556399   1.8875893   2.2475552  57.1485620
##  [97]  18.1946787   0.6482751   0.9424105  28.2363820  33.2275446   2.1626696
## [103]   1.9688031   2.2366552   1.7257163   9.5107000   6.3620228   1.5450154
## [109]   1.7647209   1.9372504   1.6180164  22.7507120   1.8139346   2.1125944
## [115]   5.7767861   1.5393408   7.9105561   1.5190788  10.1491737   2.5951164
## [121]   1.4157097   8.8474773   1.4700390   1.9213802   1.8082154   1.8936418
## [127]   1.5452155   4.2414943   1.8978162   2.1359969   1.7698750   2.0807582
## [133]   4.0484196  11.1256015  13.4319820   3.5349117   1.7952094   3.9836162
## [139]   0.4548886   4.5854895   5.0617295   1.8883696   5.6662644   1.0170599
## [145]   3.7532252   3.5521669  18.4063345   2.8271773   2.1026736   1.7571415
## [151]   3.1344558   8.6833023   1.3097698   3.0454652   1.8519723  13.1443268
## [157]   3.2924820  11.9503718   2.6679982  17.1499995   1.6696131   1.3393446
## [163]   2.0662911   4.6032481   4.3787500   1.5202907   2.3029672  23.3035719
## [169]  57.3088773   1.8648810  22.0892955   5.7263430  36.5508221   1.4069848
## [175]   2.1159839  14.0674358   1.6800708   1.6831251   1.2558438   2.4426296
## [181]   5.3075642  11.8855746  14.7338827   2.1652683   3.6391024   2.0872915
## [187]  14.7175411   2.9866214   8.0500714   1.7783108   1.7097999   7.8834240
## [193]   9.8417513   6.1738887   1.5012341   1.5229236  10.3123360   2.0687295
## [199]   3.7320430   1.2989721
```

In Non-hierarchical of K-means cluster visulization, the performance of 3 cluster is good.

```r
# K-Means Clustering
#house_data_scale <- scale(house_data[,2:10])
matstd.employ <- scale(house_data[, c(1:4,6,7:9)])
# K-means, k=2, 3, 4, 5, 6
# Centers (k's) are numbers thus, 10 random sets are chosen

(kmeans2.employ <- kmeans(matstd.employ,2,nstart = 10))
```

```
## K-means clustering with 2 clusters of sizes 155, 45
##
## Cluster means:
##         area      rooms  bathroom parking_spaces        hoa rent_amount
## 1 -0.4044422 -0.3275507 -0.400798     -0.3721702 -0.2891317  -0.4008179
## 2  1.3930786  1.1282302  1.380526      1.2819195  0.9958980   1.3805950
##   property_tax fire_insurance
## 1   -0.2549272     -0.4064196
## 2    0.8780827      1.3998898
##
## Clustering vector:
##   [1] 2 1 1 1 1 2 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1 2 1 1 2 1 2 2 2
##  [38] 1 1 1 1 2 1 1 1 2 2 1 2 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 2 2 1 1 1 1
## [112] 2 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1 2 1 2 1
## [149] 1 1 1 1 1 1 1 2 1 2 1 2 1 1 1 2 1 1 1 2 2 1 2 1 2 1 1 2 1 1 1 1 1 2 2 1 1
## [186] 1 2 1 1 1 1 1 2 2 1 1 2 1 1 1
##
## Within cluster sum of squares by cluster:
## [1] 354.9749 517.5292
##  (between_SS / total_SS =  45.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
# Computing the percentage of variation accounted for. Two clusters
perc.var.2 <- round(100*(1 - kmeans2.employ$betweenss/kmeans2.employ$totss),1)
names(perc.var.2) <- "Perc. 2 clus"
perc.var.2
```

```
## Perc. 2 clus
##         54.8
```

```r
# Computing the percentage of variation accounted for. Three clusters
(kmeans3.employ <- kmeans(matstd.employ,3,nstart = 10))
```

```
## K-means clustering with 3 clusters of sizes 28, 109, 63
##
## Cluster means:
##         area      rooms   bathroom parking_spaces         hoa rent_amount
```

```
## 1  1.8246976  1.2407568  1.5236781      1.4429141  1.37628951  1.95844068
## 2 -0.6048550 -0.7028808 -0.6737420     -0.6036744 -0.39563277 -0.54260321
## 3  0.2355184  0.6646478  0.4884903      0.4031573  0.07282326  0.06837161
##   property_tax fire_insurance
## 1   1.31827228     1.97333498
## 2  -0.29452879    -0.55461590
## 3  -0.07631724     0.08253577
##
## Clustering vector:
##   [1] 3 2 2 2 3 3 2 3 3 3 2 3 2 2 2 2 2 2 3 2 2 3 1 1 3 2 2 2 2 3 2 3 3 2 3 3 1
##  [38] 2 2 3 2 1 3 3 3 3 1 2 1 2 3 2 2 1 1 2 2 2 2 2 2 3 3 2 3 3 3 3 2 2 2 2 3 3 2
##  [75] 3 2 2 2 2 3 2 2 3 2 3 2 3 2 2 3 3 2 1 2 3 3 3 2 2 2 1 2 2 2 2 1 1 2 2 2 2
## [112] 1 2 2 2 3 1 3 1 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 3 3 2 3 2 3 3 1 2
## [149] 2 2 2 3 2 2 2 1 3 1 3 3 2 2 2 1 2 2 3 1 1 2 1 3 1 3 2 1 2 2 3 2 3 3 3 3 2
## [186] 2 1 3 3 2 3 3 1 3 2 2 1 2 2 2
##
## Within cluster sum of squares by cluster:
## [1] 378.9754 132.8573 169.3233
##  (between_SS / total_SS =  57.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
perc.var.3 <- round(100*(1 - kmeans3.employ$betweenss/kmeans3.employ$totss),1)
names(perc.var.3) <- "Perc. 3 clus"
perc.var.3
```

```
## Perc. 3 clus
##         42.8
```

```r
# Computing the percentage of variation accounted for. Four clusters
(kmeans4.employ <- kmeans(matstd.employ,4,nstart = 10))
```

```
## K-means clustering with 4 clusters of sizes 25, 63, 109, 3
##
## Cluster means:
##         area      rooms   bathroom parking_spaces         hoa rent_amount
## 1  1.7671465  1.2565240  1.5130304      1.4509639  1.12416847  2.02627717
## 2  0.2355184  0.6646478  0.4884903      0.4031573  0.07282326  0.06837161
## 3 -0.6048550 -0.7028808 -0.6737420     -0.6036744 -0.39563277 -0.54260321
## 4  2.3042896  1.1093635  1.6124085      1.3758331  3.47729817  1.39313658
##   property_tax fire_insurance
## 1   0.69080334     2.05779313
## 2  -0.07631724     0.08253577
## 3  -0.29452879    -0.55461590
## 4   6.54718016     1.26951705
##
## Clustering vector:
##   [1] 2 3 3 3 2 2 3 2 2 2 3 2 3 3 3 3 3 3 2 3 3 2 1 1 2 3 3 3 3 2 3 2 2 3 2 2 4
##  [38] 3 3 2 3 1 2 2 2 2 1 3 1 3 2 3 3 1 4 3 3 3 3 3 3 2 2 3 2 2 2 3 3 3 3 2 2 3
##  [75] 2 3 3 3 3 2 3 3 2 3 2 3 2 3 3 2 2 3 1 3 2 2 2 3 3 3 1 3 3 3 3 1 1 3 3 3 3
```

```
## [112] 1 3 3 3 2 1 2 1 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 2 2 3 2 3 2 2 1 3
## [149] 3 3 3 2 3 3 3 1 2 1 2 2 3 3 3 1 3 3 2 1 1 3 1 2 4 2 3 1 3 3 2 3 2 2 2 2 3
## [186] 3 1 2 2 3 2 2 1 2 3 3 1 3 3 3
##
## Within cluster sum of squares by cluster:
## [1] 202.61848 169.32332 132.85726  66.04712
##  (between_SS / total_SS =  64.1 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
perc.var.4 <- round(100*(1 - kmeans4.employ$betweenss/kmeans4.employ$totss),1)
names(perc.var.4) <- "Perc. 4 clus"
perc.var.4
```

```
## Perc. 4 clus
##         35.9
```

```r
# Computing the percentage of variation accounted for. Five clusters
(kmeans5.employ <- kmeans(matstd.employ,5,nstart = 10))
```

```
## K-means clustering with 5 clusters of sizes 3, 55, 39, 84, 19
##
## Cluster means:
##         area      rooms    bathroom parking_spaces        hoa rent_amount
## 1  2.3042896  1.1093635  1.61240845      1.3758331  3.4772982   1.3931366
## 2 -0.2360799  0.2346335 -0.04539769     -0.1139769 -0.2556065  -0.2381233
## 3  0.6426863  0.8699012  0.90529558      0.7798434  0.4017428   0.3831059
## 4 -0.6686386 -0.9019643 -0.82767698     -0.6785252 -0.3997821  -0.5964990
## 5  1.9564420  1.3476801  1.67778873      1.5117605  1.1336943   2.3201138
##   property_tax fire_insurance
## 1    6.5471802      1.2695170
## 2   -0.2239781     -0.2537792
## 3    0.1318520      0.3896113
## 4   -0.3065507     -0.5969742
## 5    0.6992255      2.3736999
##
## Clustering vector:
##   [1] 3 4 4 4 3 3 4 2 2 3 2 3 2 4 4 2 2 4 3 4 4 2 3 5 2 4 4 4 2 3 4 3 3 4 3 3 1
##  [38] 2 4 2 4 3 3 2 3 3 5 2 5 4 2 2 2 3 1 4 4 4 4 4 4 2 2 4 3 2 2 4 4 4 4 3 2 4
##  [75] 2 4 4 4 4 3 2 4 2 4 2 4 2 4 2 3 3 4 5 4 2 3 3 2 2 4 5 4 4 4 4 3 5 4 4 4 4
## [112] 5 4 4 2 2 3 2 5 2 4 2 4 4 4 4 4 2 2 4 4 4 2 4 5 4 4 4 2 3 3 4 3 2 3 2 5 4
## [149] 2 4 4 2 4 2 4 5 2 5 3 3 4 4 2 3 4 4 2 5 5 4 5 3 1 2 4 5 4 4 2 4 2 3 3 2 4
## [186] 4 5 2 2 2 3 3 5 3 4 4 5 2 4 4
##
## Within cluster sum of squares by cluster:
## [1]  66.04712  77.78329 128.65248  76.09341 164.50816
##  (between_SS / total_SS =  67.8 %)
##
## Available components:
##
```

```
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
perc.var.5 <- round(100*(1 - kmeans5.employ$betweenss/kmeans5.employ$totss),1)
names(perc.var.5) <- "Perc. 5 clus"
perc.var.5
```

```
## Perc. 5 clus
##         32.2
```

```r
(kmeans6.employ <- kmeans(matstd.employ,6,nstart = 10))
```

```
## K-means clustering with 6 clusters of sizes 57, 7, 12, 1, 92, 31
##
## Cluster means:
##          area      rooms   bathroom parking_spaces        hoa rent_amount
## 1 -0.08912326  0.3944139  0.1086620   -0.008155648 -0.1952276  -0.1408421
## 2  2.93038895  1.8775088  2.1092986    1.946022286 -0.7174333   1.7363929
## 3  1.70897354  1.1801137  1.4881859    1.317137111  2.6349218   2.6558063
## 4  1.51540451  1.3923644  2.1092986    2.549752055  2.2357958  -0.4852191
## 5 -0.65412488 -0.8408818 -0.7748245   -0.658109196 -0.4004236  -0.5974912
## 6  0.73302273  0.8446207  0.9792743    0.936560147  0.6172298   0.6276768
##   property_tax fire_insurance
## 1   -0.1932271     -0.1578228
## 2    0.9801326      2.1220072
## 3    1.3285123      2.4989517
## 4   10.6664667     -0.5063329
## 5   -0.3024947     -0.5968291
## 6    0.1733520      0.6312594
##
## Clustering vector:
##   [1] 6 5 5 5 6 6 5 1 1 6 5 6 5 5 5 1 1 5 1 5 5 1 2 2 1 5 5 5 5 6 5 6 6 5 6 6 3
##  [38] 1 5 1 5 6 1 1 1 6 3 5 3 5 1 5 1 6 4 5 5 5 5 5 5 1 1 5 6 1 1 5 5 5 5 1 1 5
##  [75] 1 5 5 5 5 1 1 5 1 5 1 5 1 5 1 6 1 5 2 5 1 1 6 5 5 5 3 5 5 5 5 6 6 5 5 5 5
## [112] 3 5 5 1 1 6 1 6 1 5 1 5 5 5 5 5 1 1 5 5 5 1 5 3 5 5 5 1 6 1 5 1 5 6 1 2 5
## [149] 1 5 5 1 5 1 5 3 1 3 6 6 5 5 1 6 5 5 1 2 2 5 2 6 3 1 5 6 5 5 1 5 1 6 6 1 5
## [186] 5 3 1 1 1 1 6 3 6 5 5 3 1 5 5
##
## Within cluster sum of squares by cluster:
## [1] 101.94260  57.15024 100.08177   0.00000  85.62218 106.01163
##  (between_SS / total_SS =  71.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
# Computing the percentage of variation accounted for. Six clusters
perc.var.6 <- round(100*(1 - kmeans6.employ$betweenss/kmeans6.employ$totss),1)
names(perc.var.6) <- "Perc. 6 clus"
perc.var.6
```

```
## Perc. 6 clus
##         28.3
```

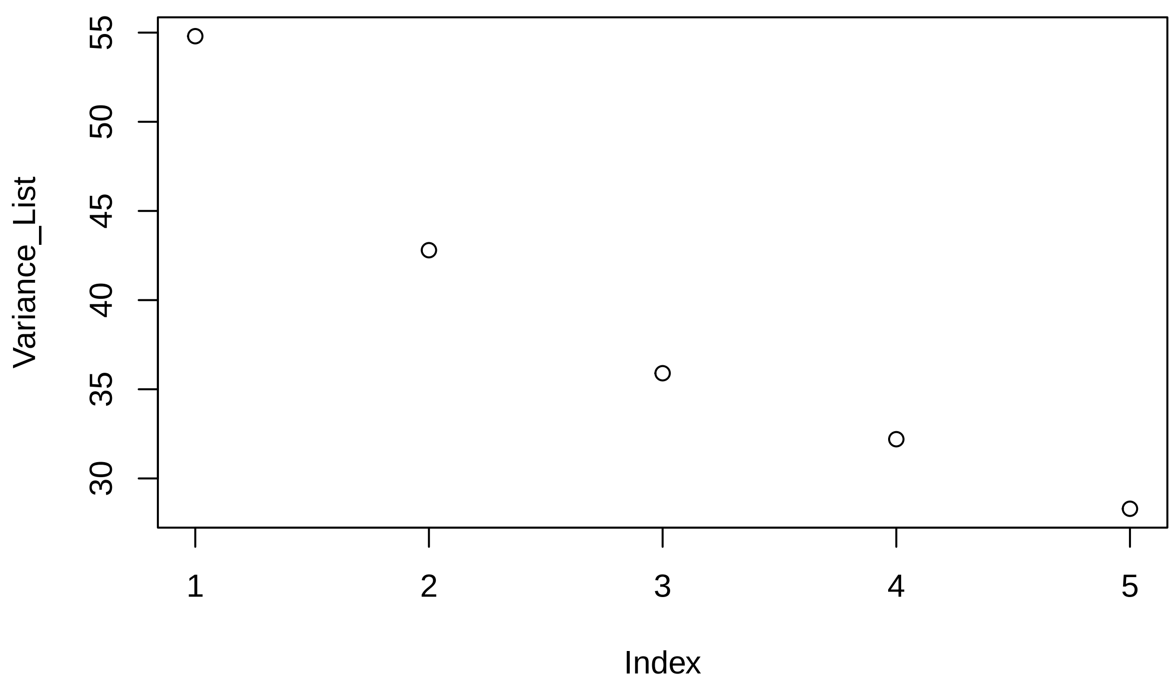
```r
attributes(perc.var.6)
```

```
## $names
## [1] "Perc. 6 clus"
```

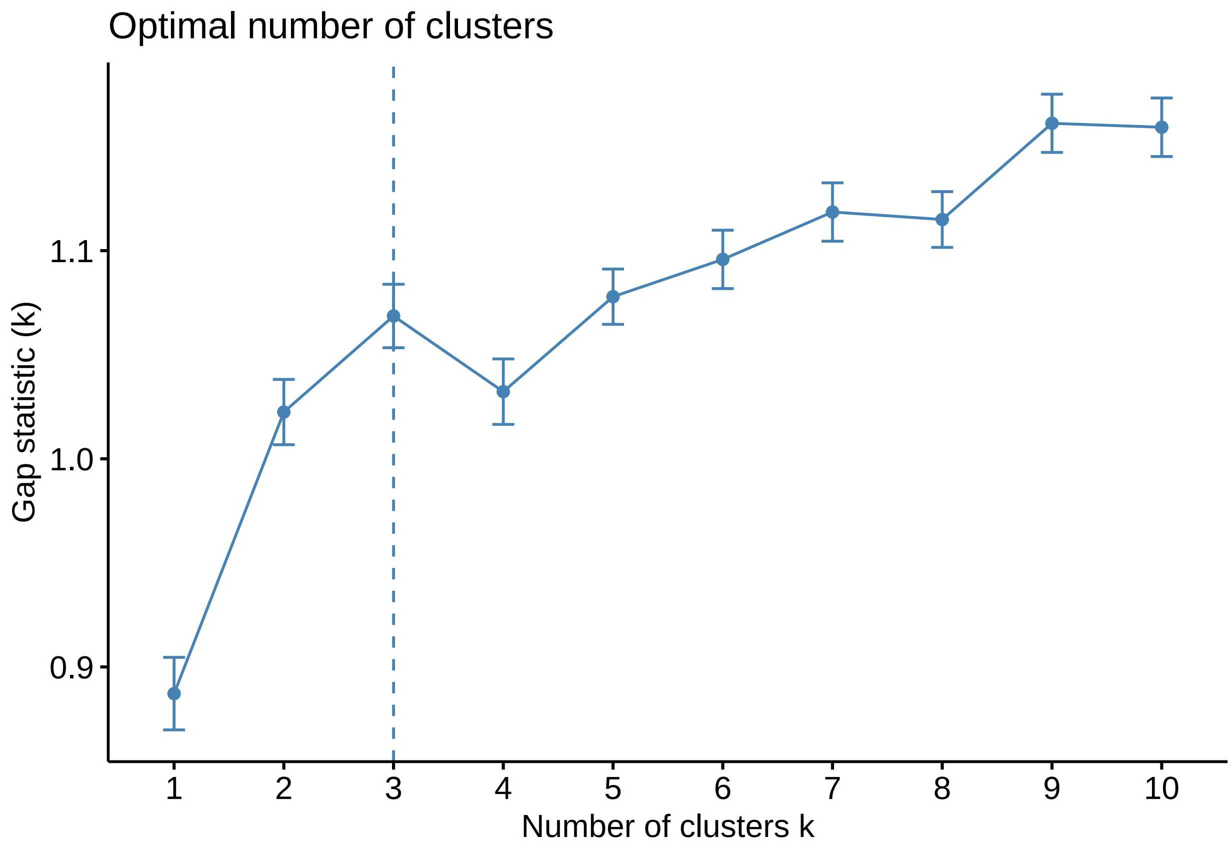
```r
Variance_List <- c(perc.var.2,perc.var.3,perc.var.4,perc.var.5,perc.var.6)

Variance_List
```

```
## Perc. 2 clus Perc. 3 clus Perc. 4 clus Perc. 5 clus Perc. 6 clus
##         54.8         42.8         35.9         32.2         28.3
```

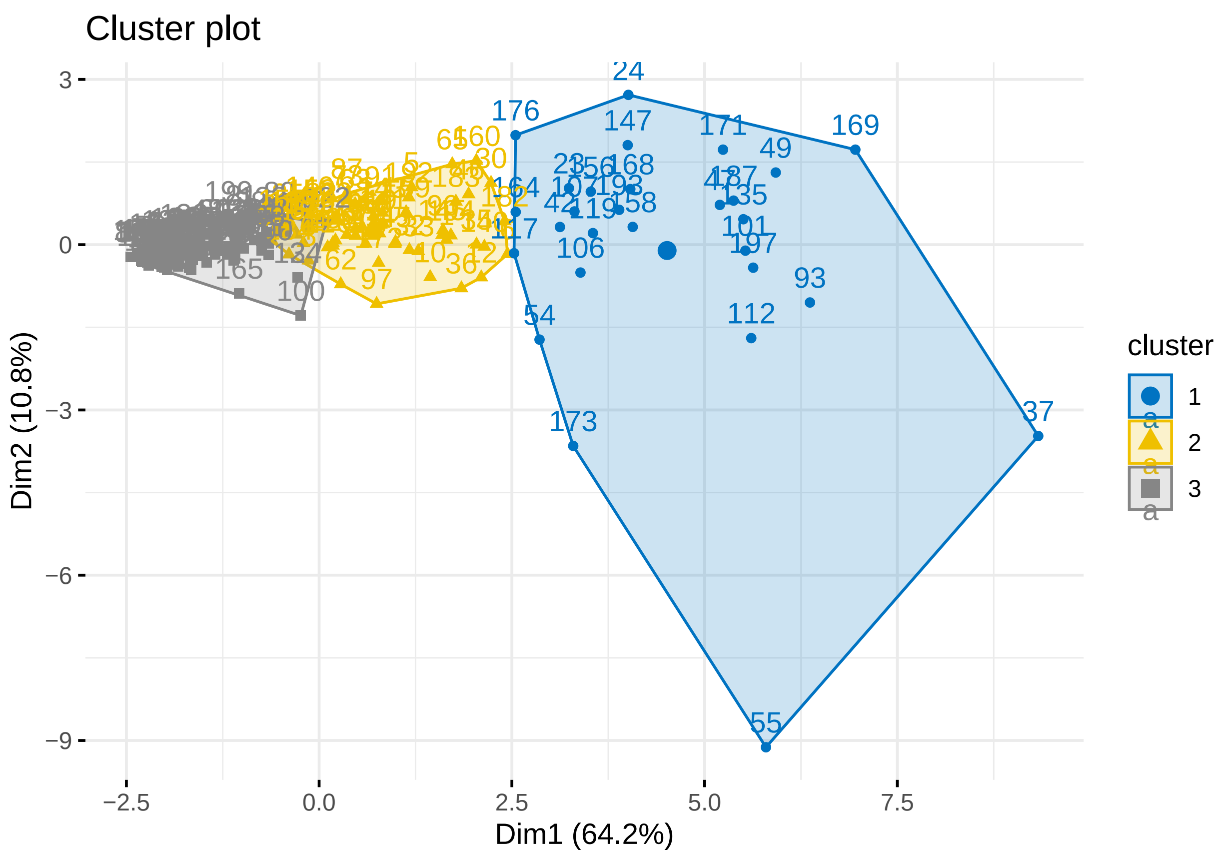
```r
plot(Variance_List)
```




```r
fviz_nbclust(matstd.employ, kmeans, method = "gap_stat")
```




```r
set.seed(123)
km.res <- kmeans(matstd.employ, 3, nstart = 25)
# Visualize
fviz_cluster(km.res, data = matstd.employ,
             ellipse.type = "convex",
             palette = "jco",
             ggtheme = theme_minimal())
```



According to the Scree diagram, we can find that there are two PCs have significant performance in percentage of explained variances. Hence, 2 components can be extracted from these variables. The visualization is as follows.

```
library(factoextra)
library(FactoMineR)
library(ggfortify)
library(psych)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(devtools)
```

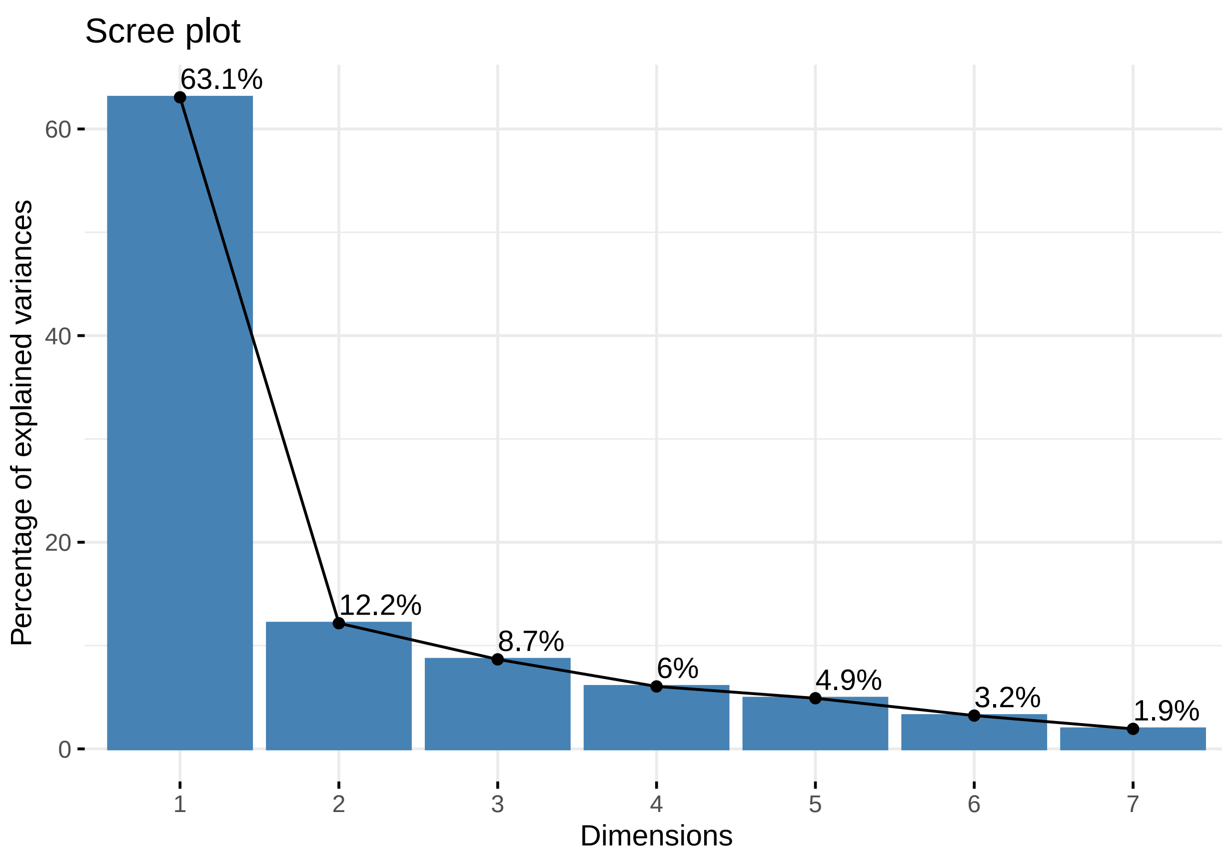
```
## Loading required package: usethis
```

```
house_PCA <- prcomp(house_data[, c(1:4,6,8:9)],scale=TRUE)
house_PCA
```

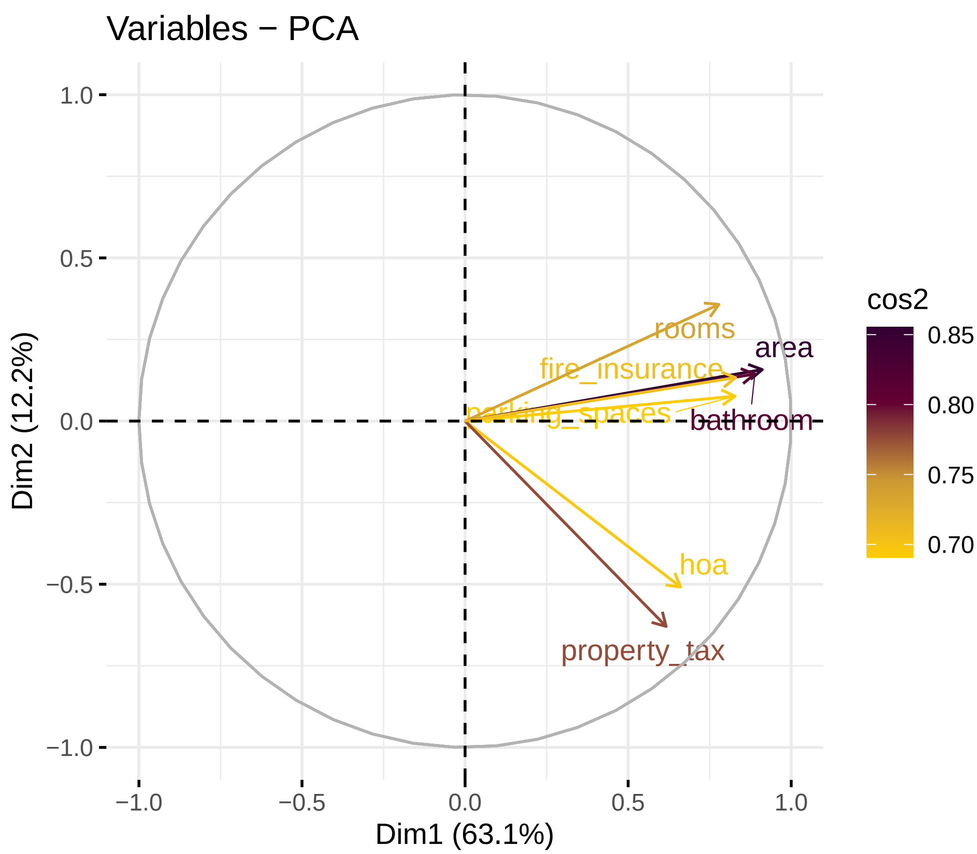
```
## Standard deviations (1, .., p=7):
## [1] 2.1011414 0.9226665 0.7787804 0.6504630 0.5857012 0.4749530 0.3683260
##
## Rotation (n x k) = (7 x 7):
##                        PC1         PC2         PC3         PC4         PC5
## area             0.4336965  0.17091900  0.11991336 -0.14262889 -0.35241854
```

```
## rooms           0.3704037  0.38742007  0.06788605  0.73084114  0.02399273
## bathroom        0.4227960  0.15626819  0.06607019  0.09708057  0.28600555
## parking_spaces 0.3939576  0.08267643  0.27820446 -0.52057449  0.56700711
## hoa             0.3144122 -0.55087277 -0.64250123  0.17481916  0.31296716
## property_tax    0.2933799 -0.68199336  0.54402406  0.12426281 -0.30542261
## fire_insurance 0.3947438  0.14443506 -0.43643932 -0.34511274 -0.52980928
##                        PC6         PC7
## area           -0.04502695  0.78850847
## rooms           0.39473933 -0.13256958
## bathroom       -0.81830269 -0.17780729
## parking_spaces  0.39883150 -0.09488344
## hoa             0.08754800  0.22068420
## property_tax    0.01872523 -0.20922784
## fire_insurance  0.07383965 -0.47705723
```

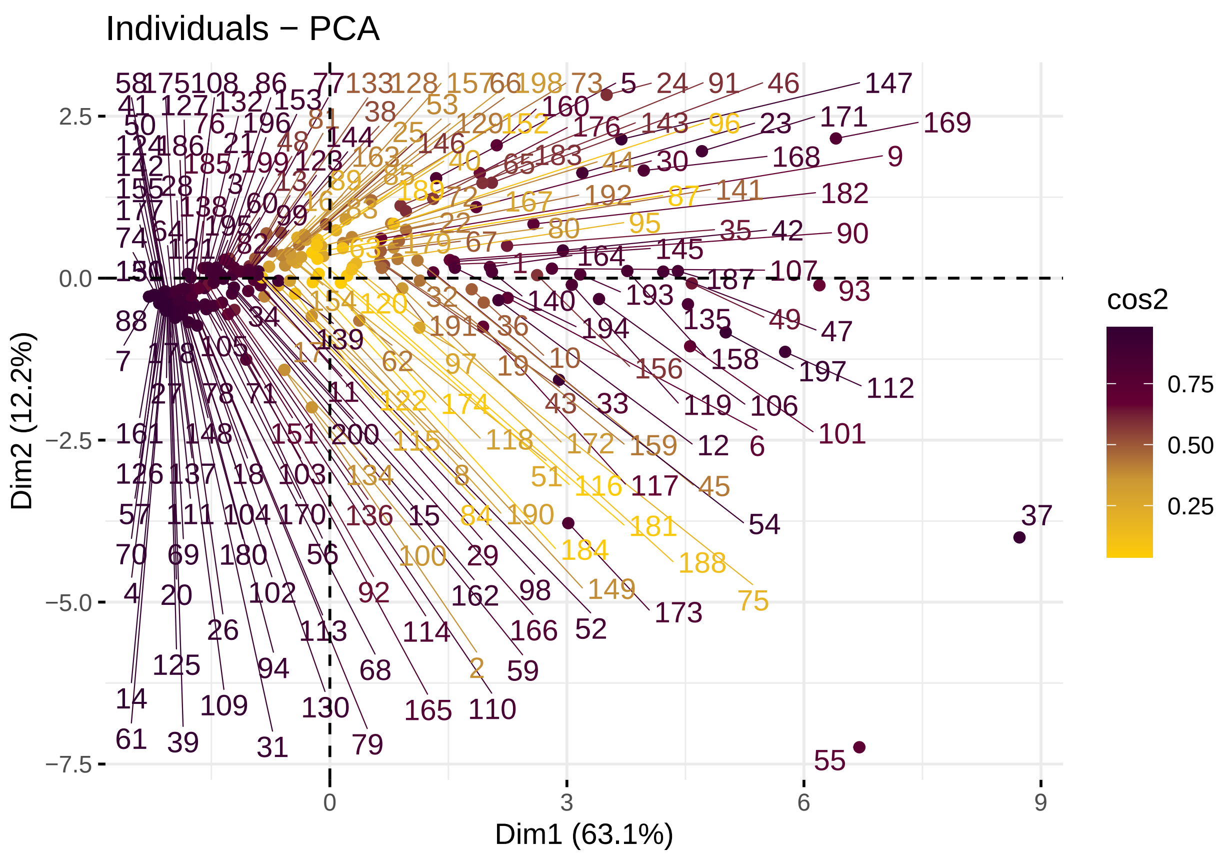
```
fviz_eig(house_PCA, addlabels = TRUE)
```



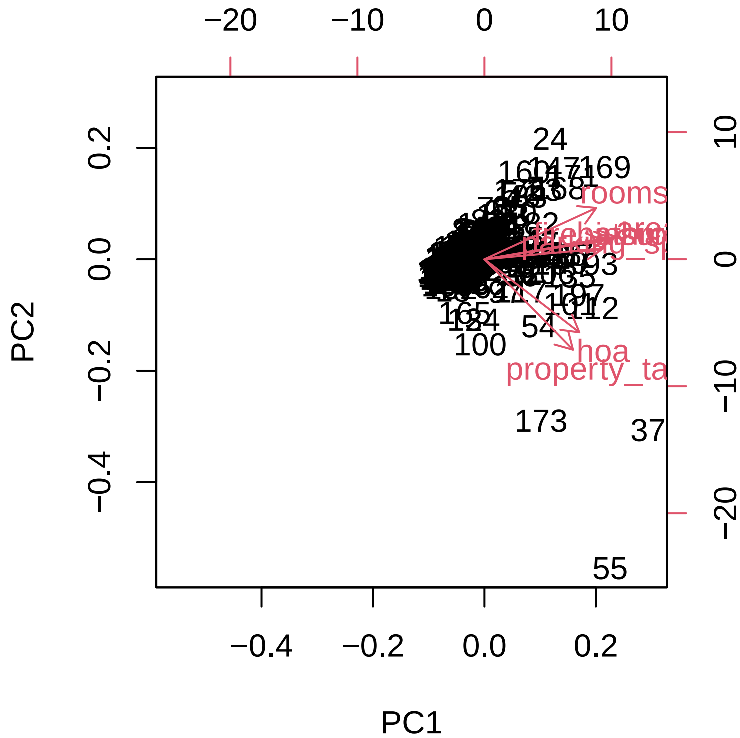

```
fviz_pca_var(house_PCA,col.var = "cos2",
             gradient.cols = c("#FFCC00", "#CC9933", "#660033", "#330033"),
             repel = TRUE)
```




```
fviz_pca_ind(house_PCA, col.ind = "cos2",
             gradient.cols = c("#FFCC00", "#CC9933", "#660033", "#330033"),
             repel = TRUE)
```

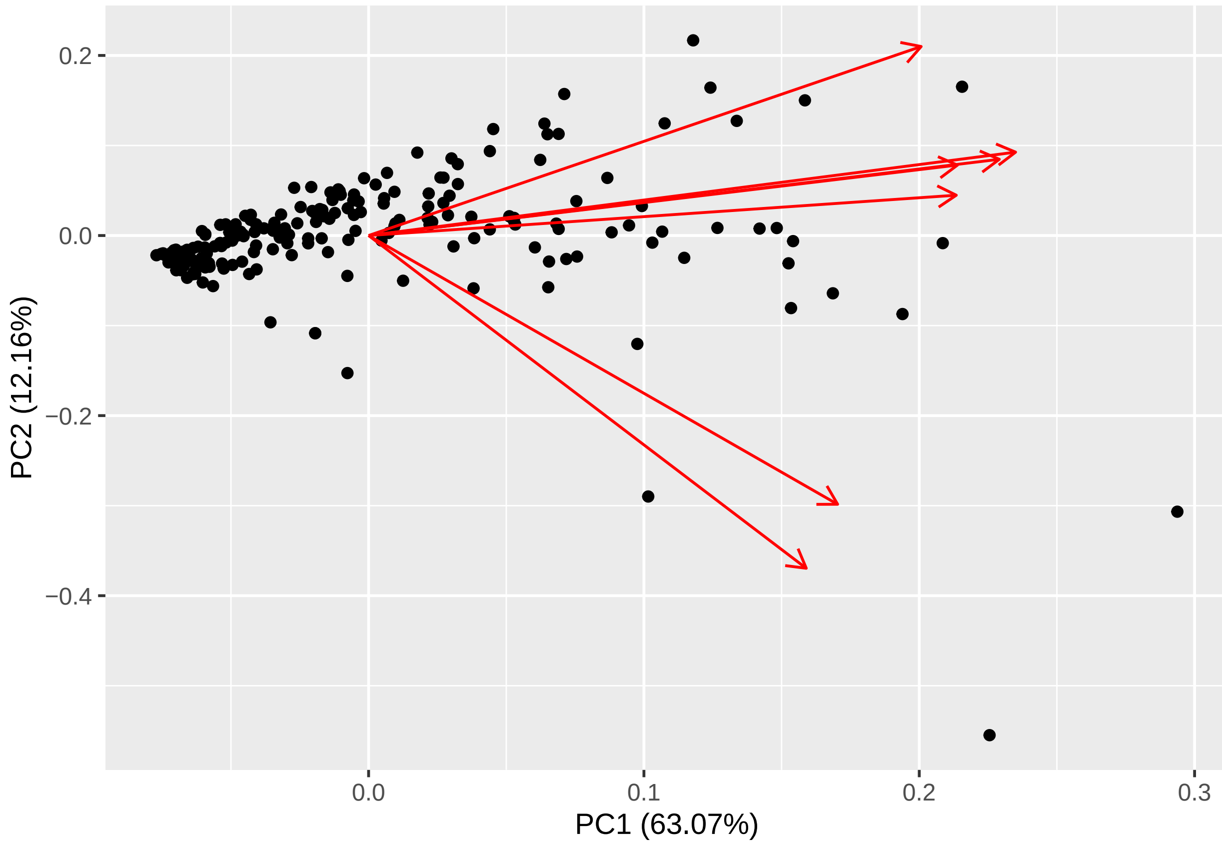



```r
biplot(house_PCA)
```




```
autoplot(house_PCA,
         data = house_data[, c(1:4,6,8:9)],
         loadings = TRUE,
         labels = house_data$Total.Score)
```

```
## Warning: Unknown or uninitialised column: `Total.Score`.
```



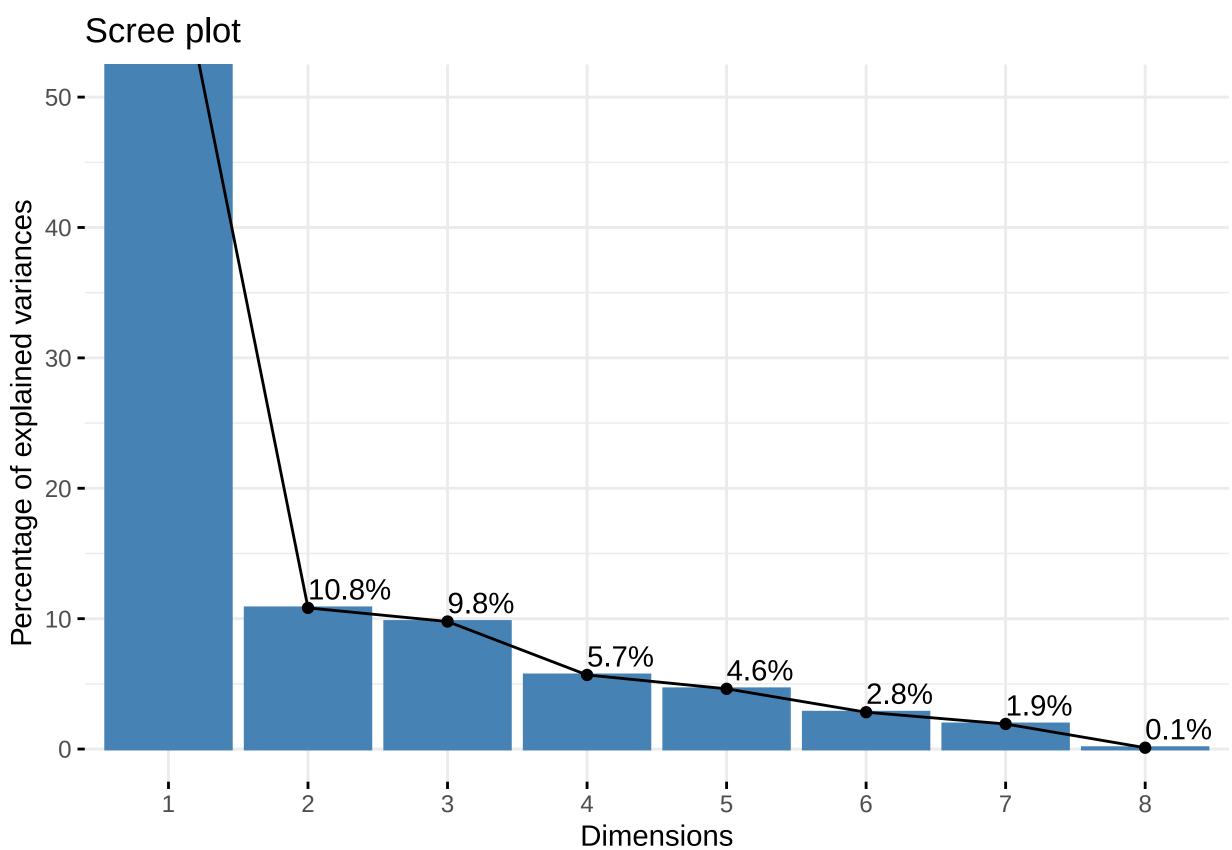

```r
res.pca <- PCA(house_data[, c(1:4,6,7:9)], graph = FALSE)
print(res.pca)
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 200 individuals, described by 8 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

```r
eig.val <- get_eigenvalue(res.pca)
eig.val
```

```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1 5.138963162       64.2370395                    64.23704
## Dim.2 0.866144721       10.8268090                    75.06385
## Dim.3 0.782491138        9.7811392                    84.84499
## Dim.4 0.454667647        5.6833456                    90.52833
## Dim.5 0.369615130        4.6201891                    95.14852
## Dim.6 0.226015504        2.8251938                    97.97372
## Dim.7 0.153891372        1.9236422                    99.89736
## Dim.8 0.008211327        0.1026416                   100.00000
```

```r
fviz_eig(res.pca, addlabels = TRUE, ylim = c(0, 50))
```




```r
var <- get_pca_var(res.pca)
var
```

```
## Principal Component Analysis Results for variables
##  ===================================================
##   Name       Description
## 1 "$coord"   "Coordinates for the variables"
## 2 "$cor"     "Correlations between variables and dimensions"
## 3 "$cos2"    "Cos2 for the variables"
## 4 "$contrib" "contributions of the variables"
```

```r
head(var$coord)
```

```
##                    Dim.1       Dim.2      Dim.3      Dim.4       Dim.5
## area           0.9047341 -0.08710863  0.1815412 -0.1253770  0.12566513
## rooms          0.7574019 -0.23177303  0.3454322  0.4295961  0.16251274
## bathroom       0.8709582 -0.04998045  0.2259705  0.1172491 -0.14147450
## parking_spaces 0.8074484  0.03564704  0.2751404 -0.2557736 -0.40600899
## hoa            0.6727193  0.37904805 -0.4825040  0.3380363 -0.21349948
## rent_amount    0.8771937 -0.21197455 -0.3828823 -0.1300784  0.09367666
```

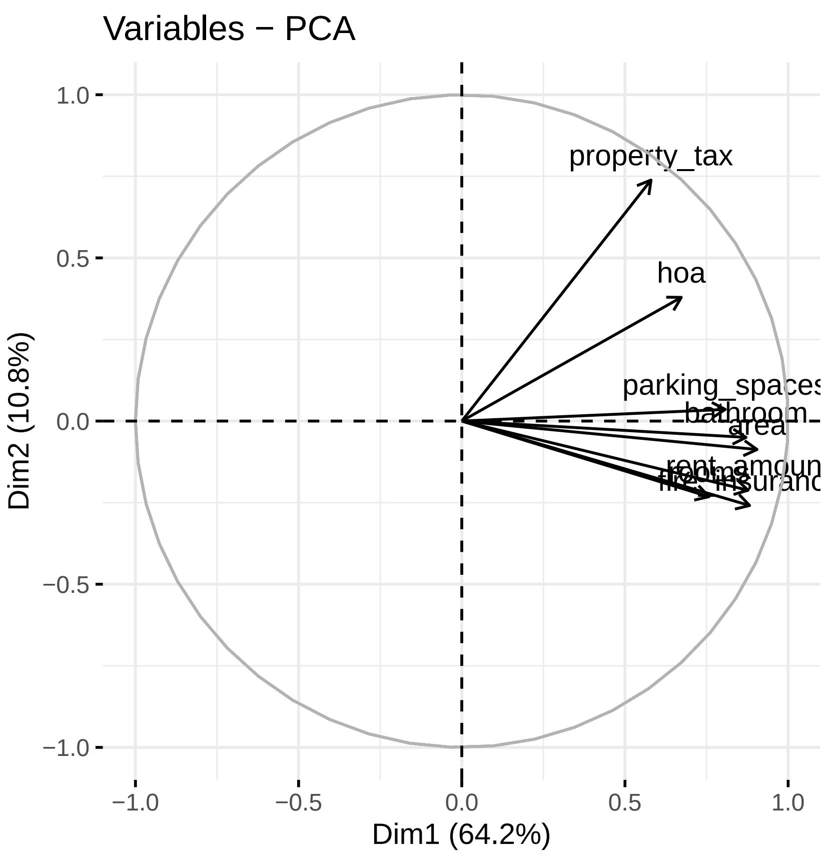
```r
head(var$cos2)
```

```
##                    Dim.1       Dim.2      Dim.3      Dim.4       Dim.5
## area           0.8185437 0.007587914 0.03295721 0.01571939 0.015791725
## rooms          0.5736577 0.053718738 0.11932343 0.18455278 0.026410391
## bathroom       0.7585683 0.002498045 0.05106266 0.01374736 0.020015035
## parking_spaces 0.6519729 0.001270712 0.07570224 0.06542012 0.164843300
## hoa            0.4525513 0.143677425 0.23281011 0.11426856 0.045582029
## rent_amount    0.7694688 0.044933211 0.14659882 0.01692040 0.008775316
```

```r
head(var$contrib)
```

```
##                    Dim.1      Dim.2     Dim.3     Dim.4     Dim.5
## area           15.928189  0.8760561  4.211832  3.457336  4.272478
## rooms          11.162907  6.2020511 15.249174 40.590701  7.145376
## bathroom       14.761115  0.2884097  6.525654  3.023606  5.415102
## parking_spaces 12.686856  0.1467089  9.674517 14.388558 44.598634
## hoa             8.806276 16.5881546 29.752428 25.132327 12.332295
## rent_amount    14.973231  5.1877255 18.734886  3.721487  2.374177
```

```r
fviz_pca_var(res.pca, col.var = "black")
```



In factor analysis, 7 variables will be transformed into 2 factors. The relationship between RC1 and "area", "rooms", "bedroom", "parking_spaces", "fire_insurance" are 0.9, 0.9, 0.8, 0.8, 0.8. The relationship between RC2 and "hoa", "property_tax"is 0.9, 0.8. The visualization is as follows.

```r
fit.pc <- principal(house_data[, c(1:4,6,8:9)], nfactors=2, rotate="varimax")
fit.pc
```

```
## Principal Components Analysis
## Call: principal(r = house_data[, c(1:4, 6, 8:9)], nfactors = 2, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                 RC1  RC2   h2   u2 com
## area           0.86 0.33 0.86 0.14 1.3
## rooms          0.85 0.09 0.73 0.27 1.0
## bathroom       0.84 0.33 0.81 0.19 1.3
## parking_spaces 0.75 0.36 0.69 0.31 1.4
## hoa            0.31 0.77 0.69 0.31 1.3
## property_tax   0.21 0.86 0.78 0.22 1.1
## fire_insurance 0.78 0.31 0.71 0.29 1.3
##
##                        RC1  RC2
## SS loadings           3.48 1.78
## Proportion Var        0.50 0.25
## Cumulative Var        0.50 0.75
## Proportion Explained  0.66 0.34
## Cumulative Proportion 0.66 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  0.08
##  with the empirical chi square  56.05  with prob <  2.8e-09
##
## Fit based upon off diagonal values = 0.98
```

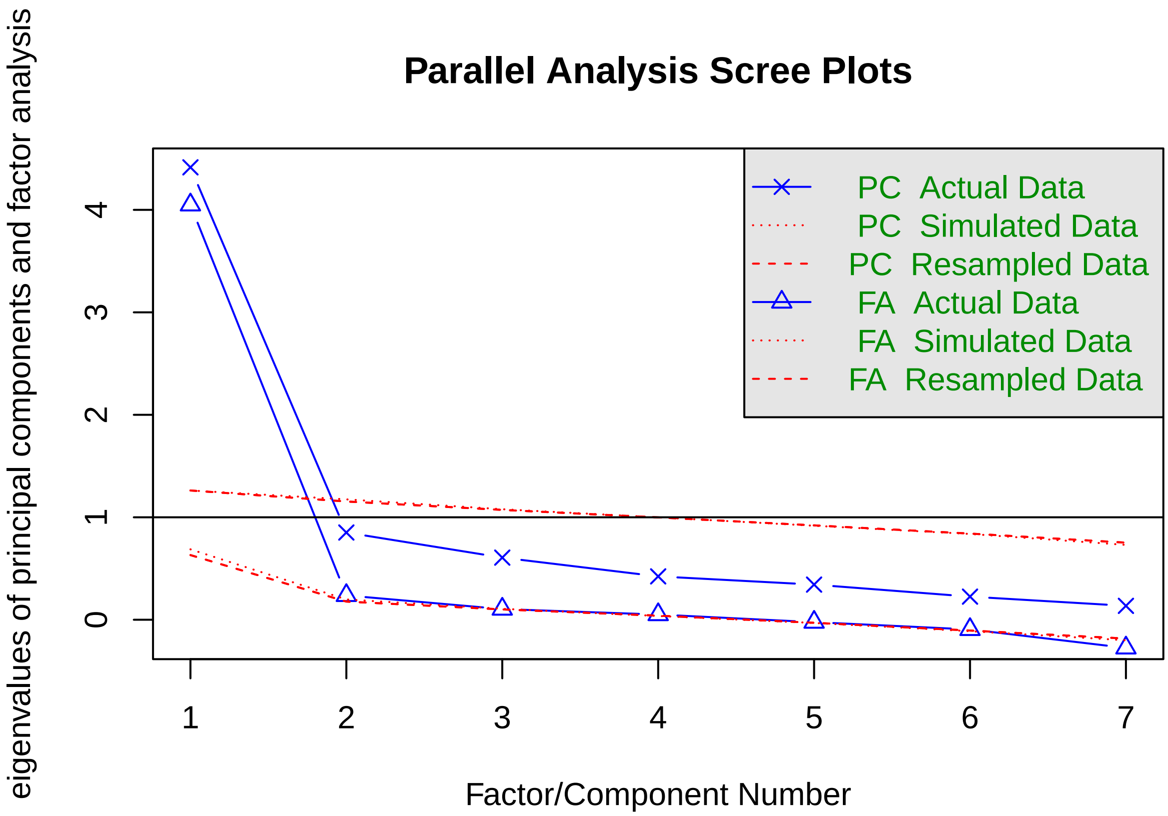
```r
round(fit.pc$values, 3)
```

```
## [1] 4.415 0.851 0.606 0.423 0.343 0.226 0.136
```

```r
fit.pc$loadings
```

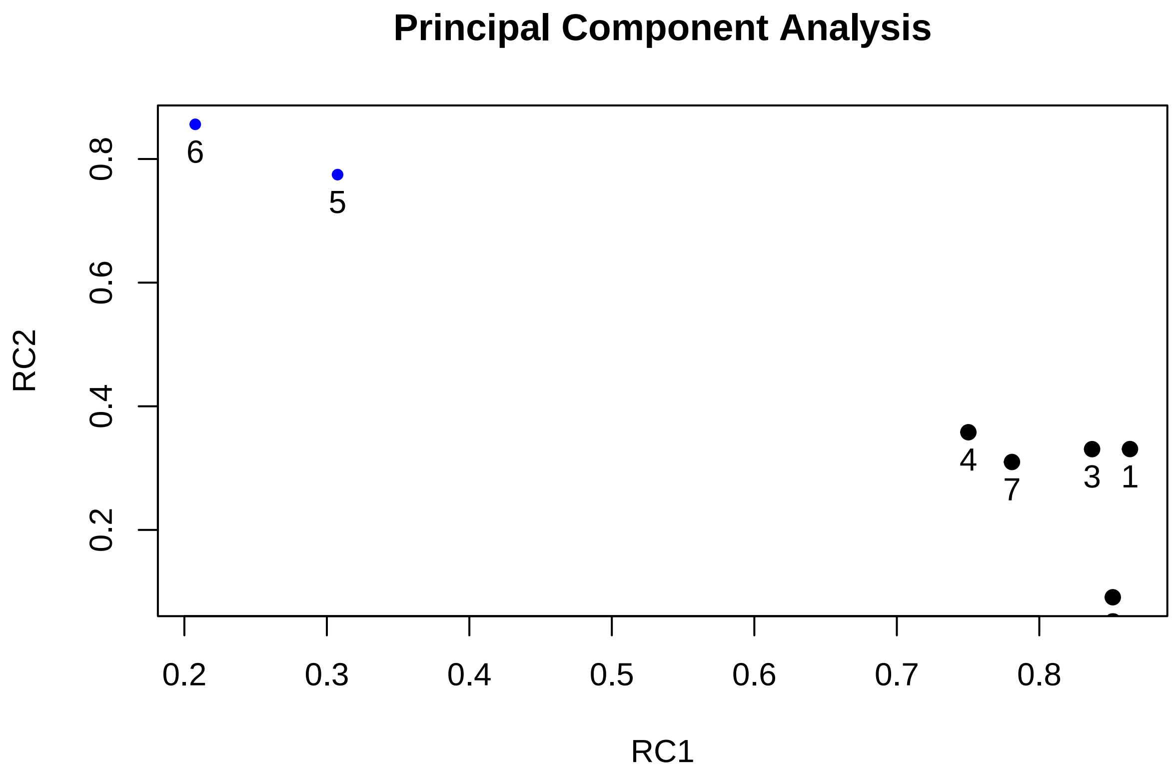
```
##
## Loadings:
##                 RC1   RC2
## area            0.864 0.331
## rooms           0.852
## bathroom        0.837 0.331
## parking_spaces 0.750 0.358
## hoa             0.308 0.775
## property_tax   0.208 0.856
## fire_insurance 0.781 0.310
##
##                   RC1   RC2
## SS loadings    3.482 1.784
## Proportion Var 0.497 0.255
## Cumulative Var 0.497 0.752
```

```r
fa.parallel(house_data[, c(1:4,6,8:9)])
```

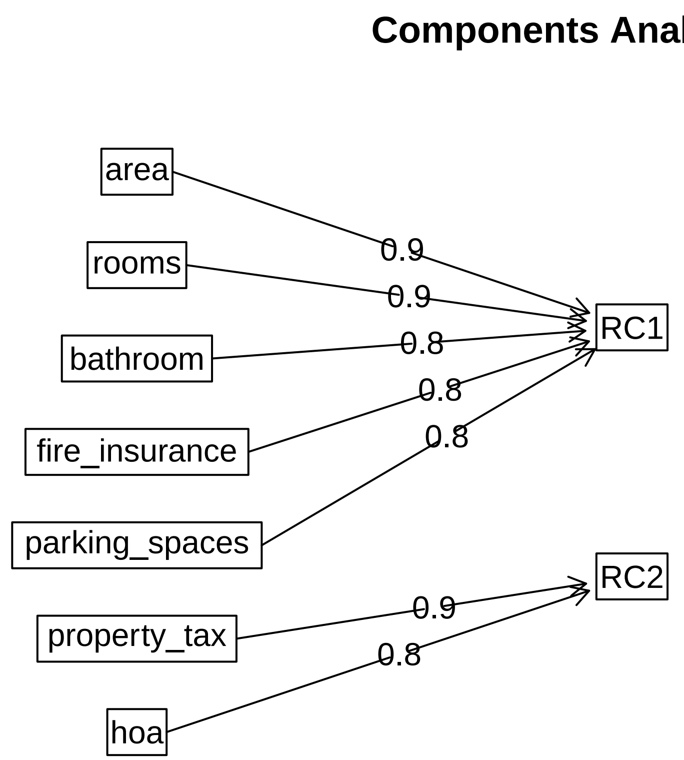



```
## Parallel analysis suggests that the number of factors =  1  and the number of components =  1
```

```r
fa.plot(fit.pc)
```



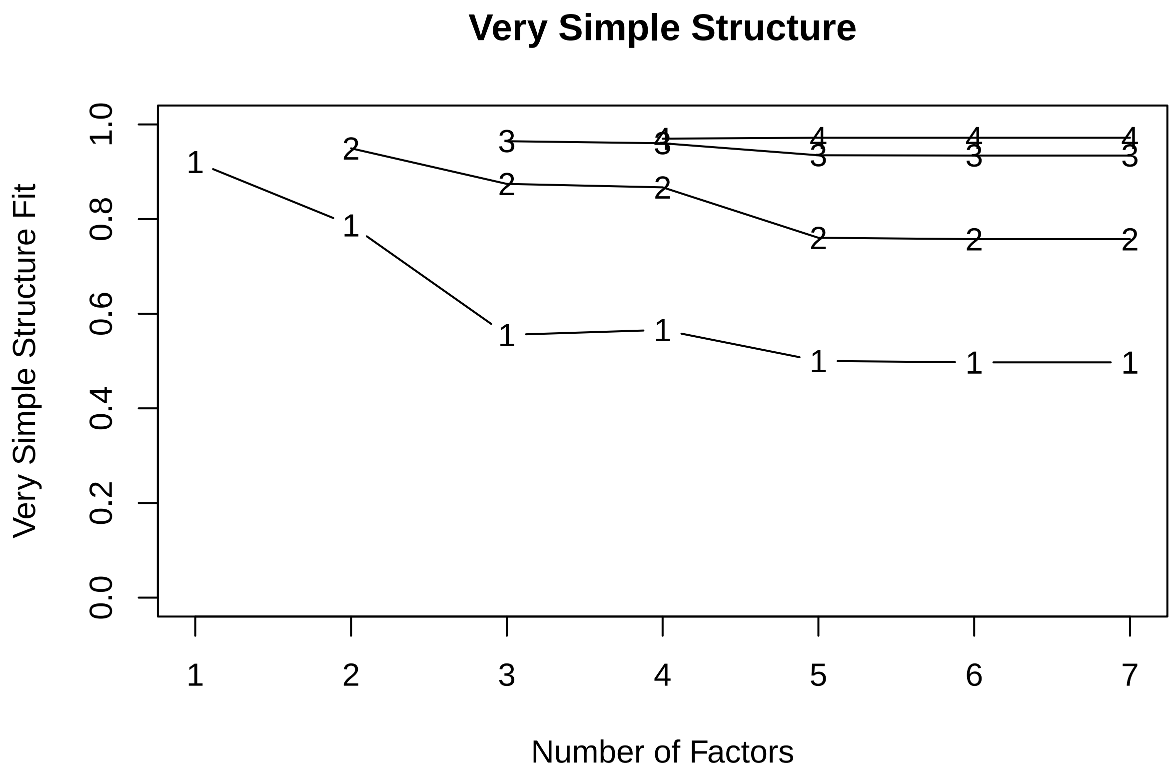

```
fa.diagram(fit.pc)
```




```r
vss(house_data[, c(1:4,6,8:9)])
```

```
## 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

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




```
##
## Very Simple Structure
## Call: vss(x = house_data[, c(1:4, 6, 8:9)])
## VSS complexity 1 achieves a maximimum of 0.92  with  1  factors
## VSS complexity 2 achieves a maximimum of 0.95  with  2  factors
##
## The Velicer MAP achieves a minimum of 0.05  with  1  factors
## BIC achieves a minimum of  7.67  with  3  factors
## Sample Size adjusted BIC achieves a minimum of  17.18  with  3  factors
##
## Statistics by number of factors
##   vss1 vss2   map dof   chisq    prob sqresid  fit RMSEA  BIC SABIC complex
## 1 0.92 0.00 0.053  14 8.6e+01 2.0e-12    1.66 0.92  0.16 12.0    56     1.0
## 2 0.79 0.95 0.102   8 5.3e+01 1.0e-08    1.06 0.95  0.17 10.8    36     1.4
## 3 0.56 0.87 0.179   3 2.4e+01 3.1e-05    0.74 0.96  0.19  7.7    17     1.7
## 4 0.57 0.87 0.306  -1 1.8e-01      NA    0.63 0.97    NA   NA    NA     1.8
## 5 0.50 0.76 0.425  -4 1.7e-10      NA    0.50 0.98    NA   NA    NA     2.1
## 6 0.50 0.76 1.000  -6 2.5e-11      NA    0.50 0.98    NA   NA    NA     2.1
## 7 0.50 0.76    NA  -7 2.5e-11      NA    0.50 0.98    NA   NA    NA     2.1
##    eChisq    SRMR eCRMS eBIC
## 1 3.1e+01 6.1e-02 0.074  -43
## 2 1.1e+01 3.6e-02 0.058  -32
## 3 3.4e+00 2.0e-02 0.053  -13
## 4 2.3e-02 1.6e-03    NA   NA
## 5 1.4e-11 4.1e-08    NA   NA
## 6 2.4e-12 1.7e-08    NA   NA
```

```
## 7 2.4e-12 1.7e-08    NA   NA
```

Question 3. Application of different MVA models (10 points)

Multiregression model

```
reg_data <- house_data[,-c(5)]
fit <- lm(rent_amount~area+rooms+bathroom+parking_spaces+hoa+property_tax+fire_insurance, data=reg_data
fit
```

```
##
## Call:
## lm(formula = rent_amount ~ area + rooms + bathroom + parking_spaces +
##     hoa + property_tax + fire_insurance, data = reg_data)
##
## Coefficients:
##    (Intercept)            area           rooms        bathroom  parking_spaces
##       54.01250        -2.49739       -67.89201        66.10065        33.40648
##            hoa    property_tax  fire_insurance
##        0.32035        -0.03111        73.32810
```

In the summary of the model, we focus on R squared value, coefficients, and P-value of each coefficient. The R-squared value is 0.9835 and Adjust R-squared value is 0.9829. It shows there is a high proportion of variance in the dependent variable can be explained by the independent variables. The result of coefficient is shown in the following table. P-value result shows that the "area", "hoa", and "fire insurance" are variables which have a significant relationship with the "rent amount" variable. In addition, we use anova to compare full model and reduced model. The result shows that only keeping "area", "hoa", and "fire insurance" does not improve the model performance. Therefore, we use stepAIC to find an optimal model. It contains "area", "rooms", "bathroom", "hoa", "fire insurance".

```
summary(fit)
```

```
##
## Call:
## lm(formula = rent_amount ~ area + rooms + bathroom + parking_spaces +
##     hoa + property_tax + fire_insurance, data = reg_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2170.3  -126.8    -9.6   109.1  4011.7
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      54.01250   76.47149   0.706  0.48085
## area             -2.49739    0.63604  -3.926  0.00012 ***
## rooms           -67.89201   41.35463  -1.642  0.10229
## bathroom         66.10065   45.03771   1.468  0.14383
## parking_spaces   33.40648   35.91696   0.930  0.35349
## hoa               0.32035    0.03715   8.623 2.42e-15 ***
## property_tax     -0.03111    0.04246  -0.733  0.46466
## fire_insurance   73.32810    1.23463  59.393  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 453.1 on 192 degrees of freedom
## Multiple R-squared:  0.9835, Adjusted R-squared:  0.9829
## F-statistic:  1634 on 7 and 192 DF,  p-value: < 2.2e-16
```

```
coefficients(fit)
```

```
##     (Intercept)            area           rooms        bathroom parking_spaces
##      54.0124954      -2.4973875     -67.8920130      66.1006537     33.4064754
##             hoa    property_tax fire_insurance
##       0.3203464      -0.0311120      73.3281041
```

```
library(MASS)
fit1 <- fit
fit2 <- lm(rent_amount ~ area + hoa + fire_insurance, data = reg_data)
# compare models
anova(fit1, fit2)
```

```
## Analysis of Variance Table
##
## Model 1: rent_amount ~ area + rooms + bathroom + parking_spaces + hoa +
##     property_tax + fire_insurance
## Model 2: rent_amount ~ area + hoa + fire_insurance
##   Res.Df      RSS Df Sum of Sq      F Pr(>F)
## 1    192 39425392
## 2    196 40564017 -4  -1138625 1.3863 0.2402
```

```
step <- stepAIC(fit, direction="both")
```

```
## Start:  AIC=2454.32
## rent_amount ~ area + rooms + bathroom + parking_spaces + hoa +
##     property_tax + fire_insurance
##
##                  Df Sum of Sq       RSS    AIC
## - property_tax    1    110224  39535616 2452.9
## - parking_spaces  1    177638  39603031 2453.2
## <none>                         39425392 2454.3
## - bathroom        1    442317  39867709 2454.6
## - rooms           1    553432  39978824 2455.1
## - area            1   3165776  42591168 2467.8
## - hoa             1  15268698  54694090 2517.8
## - fire_insurance  1 724340651 763766043 3045.1
##
## Step:  AIC=2452.88
## rent_amount ~ area + rooms + bathroom + parking_spaces + hoa +
##     fire_insurance
##
##                  Df Sum of Sq       RSS    AIC
## - parking_spaces  1    145498  39681115 2451.6
## <none>                         39535616 2452.9
```

```
## - bathroom        1    432197  39967813 2453.1
## - rooms           1    508595  40044211 2453.4
## + property_tax    1    110224  39425392 2454.3
## - area            1   3974795  43510411 2470.0
## - hoa             1  16784818  56320434 2521.7
## - fire_insurance  1 787848804 827384420 3059.1
##
## Step:  AIC=2451.61
## rent_amount ~ area + rooms + bathroom + hoa + fire_insurance
##
##                  Df Sum of Sq       RSS    AIC
## <none>                         39681115 2451.6
## - rooms           1    524159  40205274 2452.2
## + parking_spaces  1    145498  39535616 2452.9
## - bathroom        1    657291  40338406 2452.9
## + property_tax    1     78084  39603031 2453.2
## - area            1   3903457  43584571 2468.4
## - hoa             1  17055295  56736410 2521.1
## - fire_insurance  1 788294198 827975313 3057.2
```

```
fit3 <- lm(rent_amount ~ area + rooms + bathroom + hoa + fire_insurance, data = reg_data)
```

Residual Analysis Two plots are used in these residual analysis. The first plot is QQ plot. We can conclude that most of residual points are located in a straight line. It satisfies normal distribution. Simlarily, the componet + residual plots tells that each variable satisfies the normal distribution. The regression can be regarded as normal distribution.

```
confint(fit3,level=0.95)
```

```
##                       2.5 %      97.5 %
## (Intercept)     -95.5543465 203.9133827
## area             -3.6077450  -1.3634035
## rooms          -146.6132114  15.2417940
## bathroom         -7.6949583 161.2625040
## hoa               0.2443655   0.3790055
## fire_insurance   71.2507593  75.9265335
```

```
fitted(fit3)
```

```
##          1          2          3          4          5          6          7
##  5511.5579  6039.6885   681.6406  1383.9164  4154.0107  7038.4675  1205.4123
##          8          9         10         11         12         13         14
##  4510.7949  2801.7961  2374.8555  1061.6334  5011.4386  2624.1868  1352.2636
##         15         16         17         18         19         20         21
##  1127.6547   825.6313  4895.2536  2455.6685  5272.1779  1097.7289   834.4511
##         22         23         24         25         26         27         28
##  4490.7649  6566.8034 10807.7098  3097.4758  1250.5806  1690.6988   922.5070
##         29         30         31         32         33         34         35
##  1290.5269  8693.0235  2028.7218  4643.2566  4360.6289  1458.3538  3767.4580
##         36         37         38         39         40         41         42
##  1868.4123 15301.4530   816.2507  1651.3093  2004.1224  1270.5657  7457.7811
##         43         44         45         46         47         48         49
```

```
## 3449.8608  3627.2732  3423.9529  3802.9174 14342.9920  1255.3540 17677.9807
##        50         51         52         53         54         55         56
## 1401.5203  1750.0622  2018.9243  1774.8381  5446.8039  2433.8874  1133.8417
##        57         58         59         60         61         62         63
## 1313.4446  1339.1832   935.8842  2091.9363   851.7560  3448.6622  6055.1788
##        64         65         66         67         68         69         70
## 2247.1027 10188.1025  1866.4088  2611.3949  1254.8880  1974.9853   702.3652
##        71         72         73         74         75         76         77
##  964.9327  3415.8088  2855.8657  1180.5566  4354.3874  2343.5899   925.4704
##        78         79         80         81         82         83         84
## 1066.2249  3162.9495  3674.5577  1306.5155   919.0199  1992.6905  1043.8500
##        85         86         87         88         89         90         91
## 1440.2634  1101.9778  7737.6622   763.8804  2698.0348  5687.3217  2969.2884
##        92         93         94         95         96         97         98
##  579.6789  9985.8610  2387.3611  2911.1075  2567.5887  1775.5807  1846.7575
##        99        100        101        102        103        104        105
## 1241.8012  4183.5312 15283.0616  2168.4036   972.0547  1523.8742  1995.8247
##       106        107        108        109        110        111        112
## 6132.6245  9645.3327  1338.2410  1895.7396  1540.6618  1628.6900  6868.6737
##       113        114        115        116        117        118        119
## 1618.3260  2687.0499  2586.4870  3540.1317  9661.3000  2247.1525  9881.9970
##       120        121        122        123        124        125        126
## 3503.6685   894.3027  7469.7123  2874.1025  1589.7760   856.4075   704.8975
##       127        128        129        130        131        132        133
##  831.9656  2807.0680  1982.5924   704.5241   947.3628  1174.0745  1623.8423
##       134        135        136        137        138        139        140
## 6039.6885 14846.0121  4502.7869  2119.3653   820.0898  3125.2719  6047.3660
##       141        142        143        144        145        146        147
## 3839.0976  1057.2564  3272.9863  1678.2828  4989.6206  3404.2812 10029.1990
##       148        149        150        151        152        153        154
## 1005.7626  2706.7486   954.8195  4150.9097  4090.8074  1252.9744  3402.8335
##       155        156        157        158        159        160        161
##  996.0957 12692.3978  2180.9557  8965.6967  6342.4183  4730.8165   961.6378
##       162        163        164        165        166        167        168
## 1621.1194  1300.6583  8962.5088  3335.7245  2553.2451  1989.7880  7105.0819
##       169        170        171        172        173        174        175
## 12734.5496  1656.6040 12011.6440  3130.5997  8334.2638  2991.7432  1045.8071
##       176        177        178        179        180        181        182
## 10870.3632  1286.8722  1143.5071  3619.3600  1148.3880  3486.3153  4027.6544
##       183        184        185        186        187        188        189
## 3834.0908  2179.0729  2833.8547  1364.0389 14411.8397  3354.7650  1883.0096
##       190        191        192        193        194        195        196
## 2152.9957  4378.3633  5482.1623 11572.6168  6635.9162  1248.6417   767.4640
##       197        198        199        200
## 12457.1312  1082.9657  3602.8115  1974.1104
```

```
residuals(fit3)
```

```
##           1           2           3           4           5           6
##   88.442092 -519.688515   68.359449   16.083648 -714.010710 -138.467475
##           7           8           9          10          11          12
##   -5.412311 -310.794918 -101.796062 -374.855542  -61.633434   46.561430
##          13          14          15          16          17          18
## -124.186787 -152.263645  -61.654674  -17.631315 4004.746445 -147.668489
```

```
##          19          20          21          22          23          24
##  227.822100  -17.728868  165.548860 -290.764878  -66.803401 -307.709799
##          25          26          27          28          29          30
## -297.475848 -230.580552 -200.698809   47.492983  -40.526926 -493.023510
##          31          32          33          34          35          36
## -378.721778  156.743404  139.371087  141.646158  -67.457978 2631.587683
##          37          38          39          40          41          42
## -301.453003  203.749273 -151.309333  195.877594 -120.565655 -157.781097
##          43          44          45          46          47          48
##   50.139241   27.726797   76.047118  447.082609  657.007996   -5.354039
##          49          50          51          52          53          54
##  322.019271 -101.520254   49.937764 -218.924255   85.161893   53.196104
##          55          56          57          58          59          60
## -443.887365  -23.841698  -13.444561 -139.183153  -35.884170  -91.936271
##          61          62          63          64          65          66
## -151.755952 -248.662227  274.821236 -207.102652 -2188.102529  233.591217
##          67          68          69          70          71          72
##   98.605078  -54.888007  -18.985313 -152.365221   85.067345  284.191240
##          73          74          75          76          77          78
## -155.865700   19.443431  145.612614 -143.589880  -25.470391   13.775098
##          79          80          81          82          83          84
##   87.050471   25.442350   93.484491  -19.019913  437.309511  -43.849987
##          85          86          87          88          89          90
##  209.736567   38.022185  651.337792  -63.880433 -198.034819 -187.321719
##          91          92          93          94          95          96
##  530.711641  181.321088   14.138959    2.638869  -11.107505 -297.588740
##          97          98          99         100         101         102
## -575.580672  -26.757526    8.198841 -1083.531184 -283.061593 -218.403571
##         103         104         105         106         107         108
##   27.945264  -23.874217    4.175282 -132.624486  154.667298  111.758978
##         109         110         111         112         113         114
##  -95.739641  159.338227  -28.689985 -221.673707  -58.326038  112.950055
##         115         116         117         118         119         120
## -186.486990  -40.131736  338.700001   52.847540  118.003023   -3.668506
##         121         122         123         124         125         126
##   55.697333  270.287734  -74.102477   90.224017  -56.407505  -64.897482
##         127         128         129         130         131         132
##  -81.965566  342.932028  142.407647 -104.524063   52.637241   24.925525
##         133         134         135         136         137         138
##  176.157683 -519.688515  153.987947  147.213112  -19.365346 -100.089760
##         139         140         141         142         143         144
##   34.728062  -47.366011  160.902424 -107.256438  -22.986312  -28.282791
##         145         146         147         148         149         150
##   10.379416 -404.281222 -1029.199014   -5.762622   93.251355 -119.819482
##         151         152         153         154         155         156
##   -0.909698  -90.807429   47.025568  297.166490 -116.095663  307.602210
##         157         158         159         160         161         162
##  -80.955712 -165.696737  157.581704 -130.816472 -161.637786   93.880615
##         163         164         165         166         167         168
##  -20.658349  537.491164 -345.724512   46.754903  160.211988   94.918107
##         169         170         171         172         173         174
##  265.450425  -56.604012 -411.644041  -30.599708  165.736180   58.256838
##         175         176         177         178         179         180
##  -45.807148 -470.363188   43.127806  -93.507089 -219.360043  -48.388027
```

```
##          181          182          183          184          185          186
##    63.684654    72.345551    65.909170  -129.072871  -113.854663  -164.038895
##          187          188          189          190          191          192
##   588.160340  -354.764978   116.990406   -52.995710   121.636653   517.837746
##          193          194          195          196          197          198
##   427.383213  -785.916150   -98.641702    52.535969    42.868823   117.034259
##          199          200
##   397.188508    15.889605
```

```r
library(car)
```

```
## Loading required package: carData
```

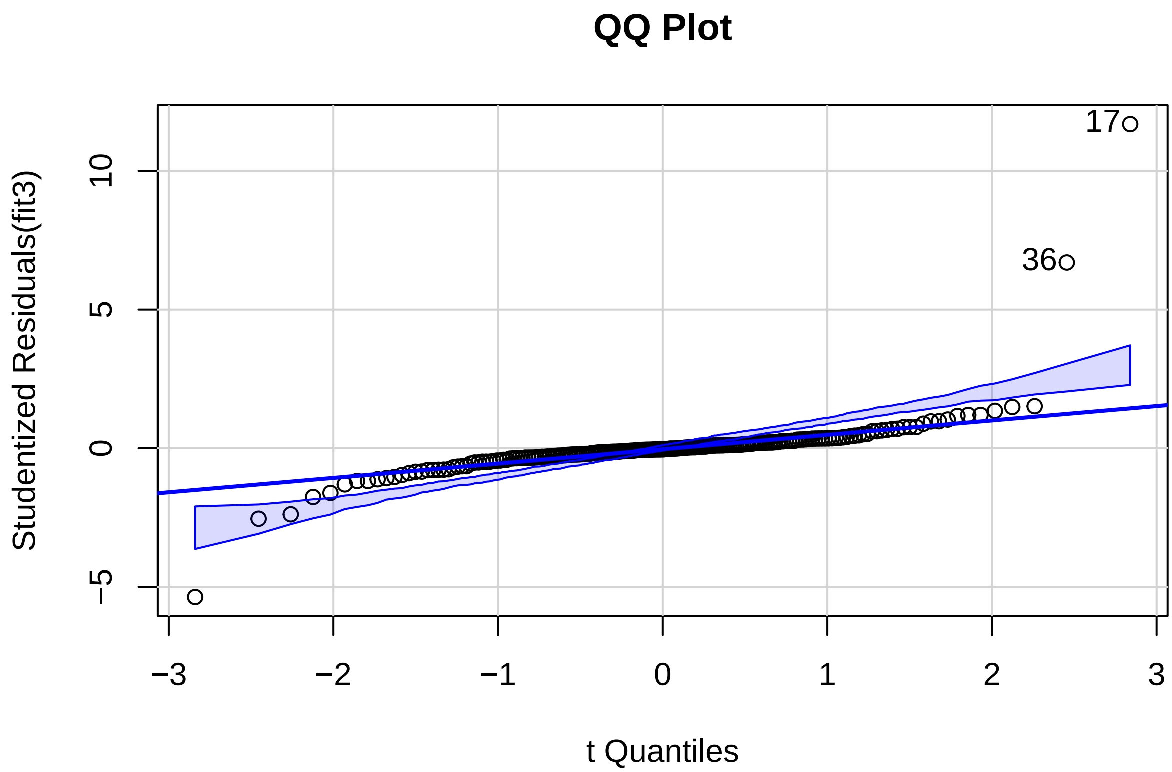
```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:psych':
##
##     logit
```

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

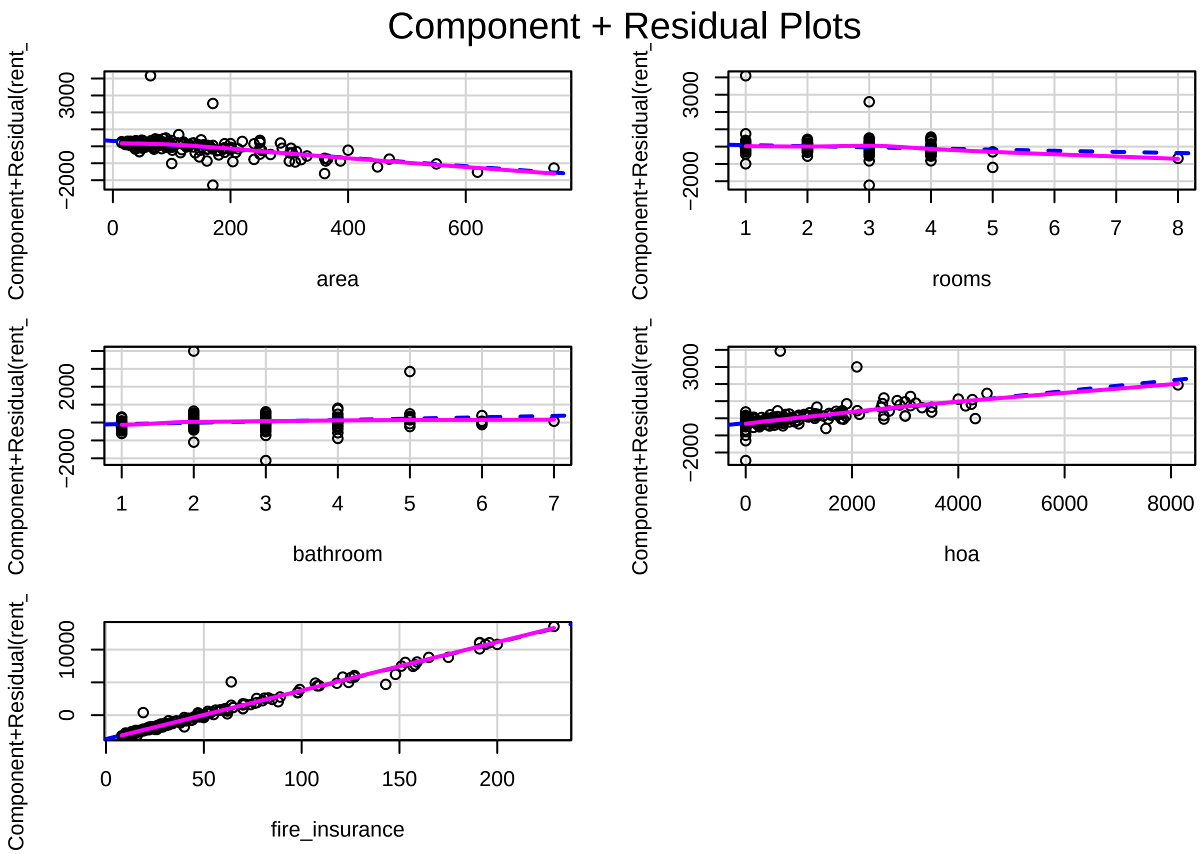
```
## The following object is masked from 'package:memisc':
##
##     recode
```

```r
qqPlot(fit3, main="QQ Plot")
```




```
## [1] 17 36
```

```
crPlots(fit3)
```



Prediction We set a data point with area = 120, rooms = 3, bathroom = 2, hoa = 0, fire insurance = 50, then the rent amount we predict is 3391.853.

```
predict.lm(fit3, data.frame(area = 120, rooms = 3, bathroom = 2, hoa = 0, fire_insurance = 50))
```

```
##        1
## 3391.853
```

Model Accuracy The accuracy is based on summary of the model and we also calculate the MSE and RMSE for the model. The R-squared value is 0.9834, and the Adjusted R-squared value is 0.983. The MSE is 198405.6 and RMSE is 445.4274.

```
summary(fit3)
```

```
##
## Call:
## lm(formula = rent_amount ~ area + rooms + bathroom + hoa + fire_insurance,
##     data = reg_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2188.1  -129.5   -17.7    94.1  4004.7
##
```

```
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     54.17952   75.91965   0.714   0.4763
## area            -2.48557    0.56897  -4.369 2.04e-05 ***
## rooms          -65.68571   41.03272  -1.601   0.1110
## bathroom        76.78377   42.83330   1.793   0.0746 .
## hoa              0.31169    0.03413   9.131  < 2e-16 ***
## fire_insurance  73.58865    1.18538  62.080  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 452.3 on 194 degrees of freedom
## Multiple R-squared:  0.9834, Adjusted R-squared:  0.983
## F-statistic:  2296 on 5 and 194 DF,  p-value: < 2.2e-16
```

```
predictions <- predict(fit3, reg_data)
mse <- mean((reg_data$rent_amount - predictions)^2)
rmse <- sqrt(mse)
cat("MSE: ", mse, "\n")
```

```
## MSE:  198405.6
```

```
cat("RMSE: ", rmse, "\n")
```

```
## RMSE:  445.4274
```

Logistics model

Running the following code, we build a multiple regression model based on rent house data. Its independent variables "area", "rooms", "bathroom", "parking spaces", "hoa", "property tax", "fire insurance". The dependent variable is "furniture".

```
reg_data <- house_data
reg_data$furniture <- as.factor(reg_data$furniture)
logistic <- glm(furniture~rooms+bathroom+area+rent_amount+property_tax, data=reg_data, family="binomial
summary(logistic)
```

```
##
## Call:
## glm(formula = furniture ~ rooms + bathroom + area + rent_amount +
##     property_tax, family = "binomial", data = reg_data)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.459e+00  4.726e-01   3.087 0.002019 **
## rooms         5.615e-01  3.103e-01   1.809 0.070415 .
## bathroom     -3.747e-01  2.550e-01  -1.469 0.141727
## area          5.310e-03  4.543e-03   1.169 0.242468
## rent_amount  -3.126e-04  8.897e-05  -3.514 0.000442 ***
## property_tax  9.175e-04  7.725e-04   1.188 0.234948
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 185.49  on 199  degrees of freedom
## Residual deviance: 163.21  on 194  degrees of freedom
## AIC: 175.21
##
## Number of Fisher Scoring iterations: 6
```

In the summary of the model, we focus on R squared value, coefficients, and P-value of each coefficient. The R-squared value is 0.1201421 and p value is 0.0004620982. The "rent amount" is a significant independent variable in this model. The Pseudo R-square shows there is an improvement and better than baseline model. The p-value based on Likelihood Ratio Test shows that the model improvement is significant. The AIC value is 175.21.

```r
summary(logistic)
```

```
##
## Call:
## glm(formula = furniture ~ rooms + bathroom + area + rent_amount +
##     property_tax, family = "binomial", data = reg_data)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.459e+00  4.726e-01   3.087 0.002019 **
## rooms         5.615e-01  3.103e-01   1.809 0.070415 .
## bathroom     -3.747e-01  2.550e-01  -1.469 0.141727
## area          5.310e-03  4.543e-03   1.169 0.242468
## rent_amount  -3.126e-04  8.897e-05  -3.514 0.000442 ***
## property_tax  9.175e-04  7.725e-04   1.188 0.234948
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 185.49  on 199  degrees of freedom
## Residual deviance: 163.21  on 194  degrees of freedom
## AIC: 175.21
##
## Number of Fisher Scoring iterations: 6
```

```r
ll.null <- logistic$null.deviance/-2
ll.proposed <- logistic$deviance/-2
(ll.null - ll.proposed) / ll.null
```

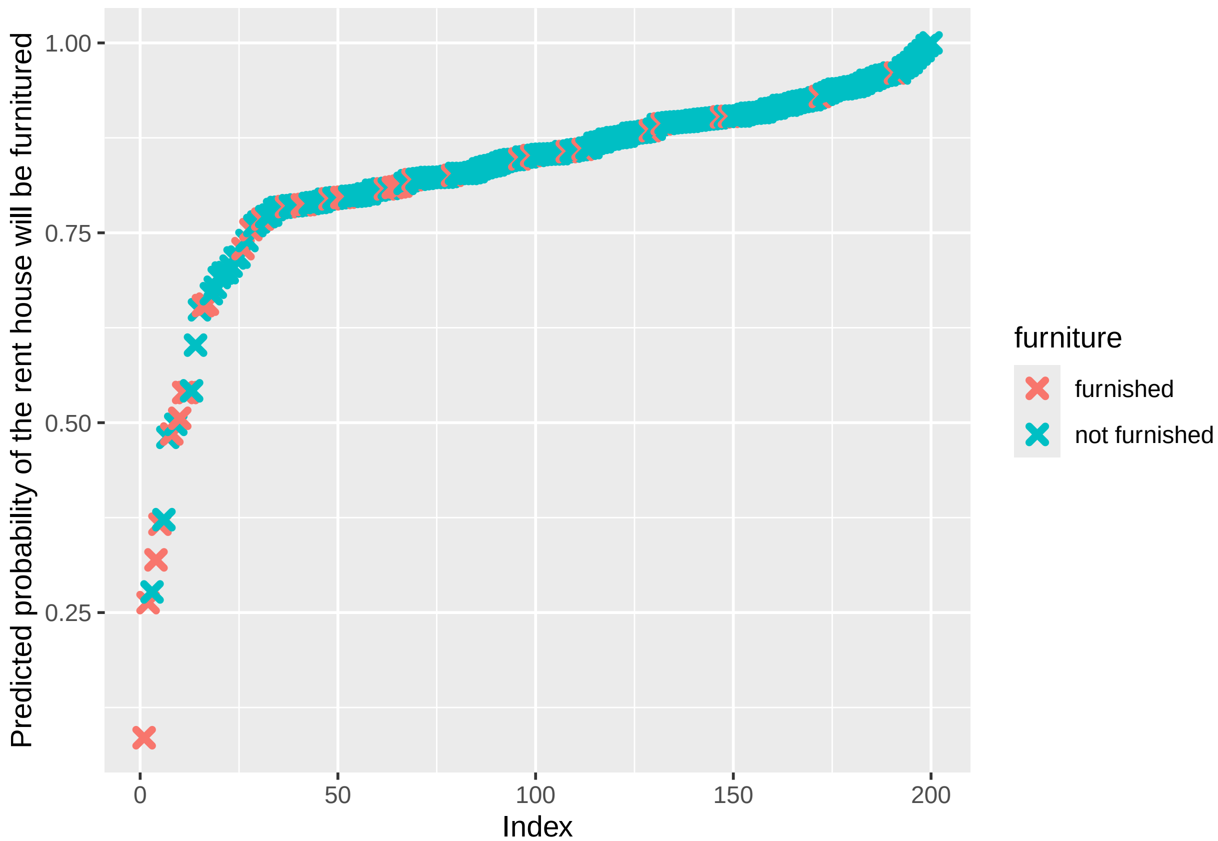
```
## [1] 0.1201421
```

```r
1 - pchisq(2*(ll.proposed - ll.null), df=(length(logistic$coefficients)-1))
```

```
## [1] 0.0004620982
```

The data will be predicted in the model and the predicted probability of the rent house will be furnitured table is as follows.

```
predicted.data <- data.frame(probability.of.furniture=logistic$fitted.values,furniture=reg_data$furnitu
predicted.data <- predicted.data[order(predicted.data$probability.of.furniture, decreasing=FALSE),]
predicted.data$rank <- 1:nrow(predicted.data)
ggplot(data=predicted.data, aes(x=rank, y=probability.of.furniture)) +
geom_point(aes(color=furniture), alpha=1, shape=4, stroke=2) +
xlab("Index") +
ylab("Predicted probability of the rent house will be furnitured")
```



To observe the performance of the model, the test set is used to approximate accuracy.

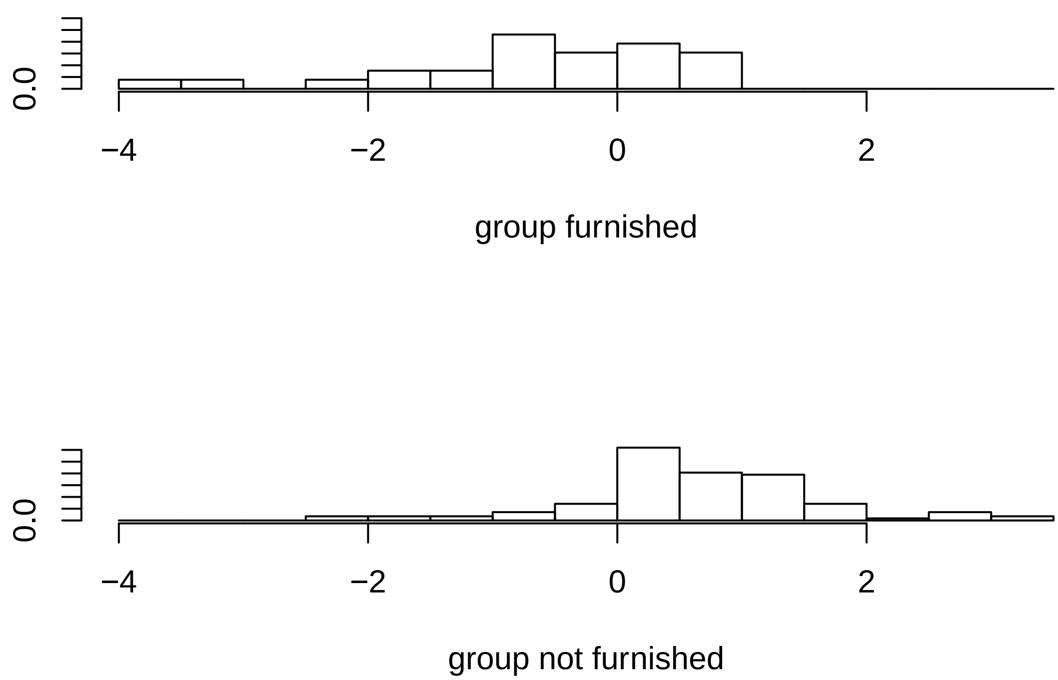
```
set.seed(101)
sample_n(house_data,10)
```

```
## # A tibble: 10 x 9
##     area rooms bathroom parking_spaces furniture    hoa rent_amount property_tax
##    <dbl> <dbl>    <dbl>          <dbl> <chr>      <dbl>       <dbl>        <dbl>
##  1   250     4        2              1 not furni~     0        2700          209
##  2    35     1        1              0 not furni~   270        1300            0
##  3    96     3        2              1 not furni~  1122        3050          231
##  4   137     3        3              1 furnished   1180        2900          214
##  5    40     1        1              1 not furni~     0        1200            0
##  6   301     4        5              4 furnished   4265       12500         1600
```

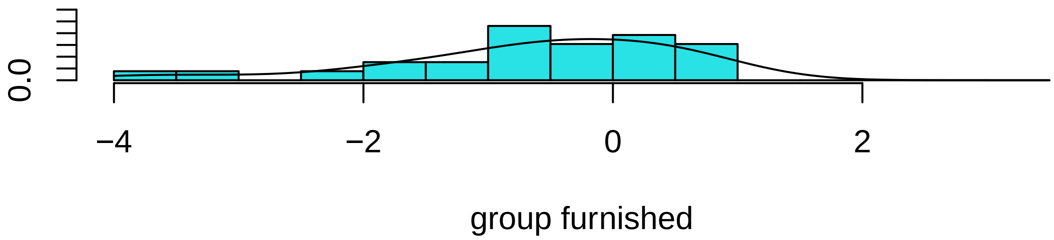
```
## 7     48      1         1               0 not furni~    309         700             28
## 8    140      2         3               2 furnished    1000        3000            113
## 9     70      2         1               1 not furni~    729         900            122
## 10    60      2         2               1 not furni~    440        1250             38
## # i 1 more variable: fire_insurance <dbl>
```

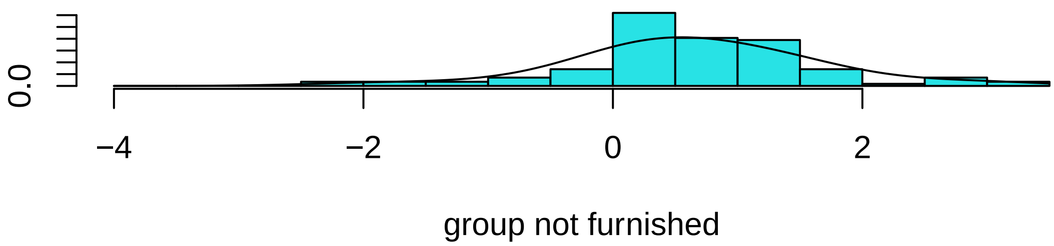
```r
training_sample <- sample(c(TRUE, FALSE), nrow(house_data), replace = T, prob = c(0.75,0.25))
train <- house_data[training_sample, ]
test <- house_data[!training_sample, ]
lda.house <- lda(furniture ~ ., train)
plot(lda.house, col = as.integer(train$furniture))
```

```
## Warning in rect(breaks[-n], 0, breaks[-1L], est[[grp]], col = col, ...): NAs
## introduced by coercion
```




```r
# Sometime bell curves are better
plot(lda.house, dimen = 1, type = "b")
```






```r
lda.train <- predict(lda.house)
train$lda <- lda.train$class
table(train$lda,train$furniture)
```

```
##
##                  furnished not furnished
##   furnished              6             6
##   not furnished         20           107
```

```r
# running accuracy on the training set shows how good the model is. It is not an indication of "true" a
lda.test <- predict(lda.house,test)
test$lda <- lda.test$class
table(test$lda,test$furniture)
```

```
##
##                  furnished not furnished
##   furnished              1             5
##   not furnished          8            47
```

Question 4 Model Insights (10 points)

It reveals that extensive utilization of various multivariate analysis (MVA) models to explore relationships between house rental prices and associated features. The analyses incorporated multiple regression models and k-means clustering to interpret the underlying structure of the dataset. Notably, we applied principal component analysis (PCA) and factor analysis to reduce dimensionality and uncover latent variables influencing rental prices.

Significant findings from the regression analysis included the identification of key predictors such as area, number of rooms, and presence of amenities like parking and fire insurance, which were significantly related to rental costs. K-means clustering demonstrated distinct groupings within the data, suggesting variations in housing characteristics that could affect rental prices.

Question 5 Learnings and Takeaways (20 points)

Learnings: The most important thing learned is diversity in model evaluation: By evaluating model performance using different statistical metrics such as R-squared, adjusted R-squared, and F-statistics, you can gain a comprehensive understanding of the model's explanatory and predictive power. This helps in selecting the most suitable model for prediction or classification. Take aways: The necessity for careful selection of variables in model building, as shown by the stepwise regression outcomes. The application of clustering methods can reveal hidden patterns and segments within the data, which are crucial for targeted marketing and investment strategies in real estate. PCA and factor analysis are powerful tools for reducing complexity in data, allowing easier interpretation and visualization.