

Fokomponenselemzés és exploratoros faktorelemzés

Zoltan Kekecs

30 november 2019

Contents

1	Absztrakt	2
2	Adat kezelés és leíró statisztikák	2
2.1	Csomagok betöltése	2
2.2	Saját funkciók betöltése	2
2.3	A cars beépített adatbázis betöltése	4
2.4	Az adatsor megtekintése	4
3	Fokomponenselemzés	8
3.1	A fokomponenselemzés modell megépítése	8
3.2	Hány új dimenziót generaljunk?	14
3.3	Fokomponenselemzés eredményeinek értelmezése	19
4	Bevezetés a feltárási faktorelemzésbe (Exploratory Factor Analysis - EFA)	28
4.1	Új adatok	29
4.2	Adatok faktorálhatósága	29
4.3	Faktorextrakció	30
4.4	Ideális faktorszám kiválasztása	32
4.5	Faktorforgatás	36
4.6	Faktorok interpretációja	36

1 Absztrakt

Ebben a gyakorlatban a “dimenzionalitas atkaval” birkozunk meg. Ezt a valtozok szamanak csokkentesevel oldjuk meg a fokomponenselemzes es az exploratoros faktorelemzes segitsegevel.

Ez a gyakorlat nagy mertekben a DataCamp Dimensionality Reduction in R kurzusa alapjan lett összeallitva. <https://www.datacamp.com/courses/dimensionality-reduction-in-r>

2 Adat kezeles és leiro statisztikak

2.1 Csomagok betoltese

Ennek a gyakorlatnak a soran az alabbi csomagokat fogjuk hasznalni:

```
library(tidyverse) # for tidy code
library(GGally)    # for ggcorr
library(corr)      # network_plot
library(ggcorrplot) # for ggcorrplot
library(FactoMineR) # multiple PCA functions
library(factoextra) # visualisation functions for PCA (e.g. fviz_pca_var)
library(paran)     # for paran

library(psych) # for the mixedCor, cortest.bartlett, KMO, fa functions
library(GPArotation) # for the psych fa function to have the required rotation functionalities
library(MVN) # for mvn function
library(ICS) # for multivariate skew and kurtosis test
```

2.2 Saját funkciók betoltese

```
fviz_loadings_with_cor <- function(mod, axes = 1, loadings_above = 0.4) {
  require(factoextra)
  require(dplyr)
  require(ggplot2)

  if (!is.na(as.character(mod$call$call)[1])) {
    if (as.character(mod$call$call)[1] == "PCA") {
      contrib_and_cov = as.data.frame(rbind(mod[["var"]][["contrib"]],
      mod[["var"]][["cor"]]))

      vars = rownames(mod[["var"]][["contrib"]])
      attribute_type = rep(c("contribution", "correlation"),
      each = length(vars))
      contrib_and_cov = cbind(contrib_and_cov, attribute_type)
      contrib_and_cov

      plot_data = cbind(as.data.frame(cbind(contrib_and_cov[contrib_and_cov[,
      "attribute_type"] == "contribution", axes], contrib_and_cov[contrib_and_cov[,
      "attribute_type"] == "correlation", axes])),
      vars)
      names(plot_data) = c("contribution", "correlation",
      "vars")
    }
  }
}
```

```

plot_data = plot_data %>% mutate(correlation = round(correlation,
2))

plot = plot_data %>% ggplot() + aes(x = reorder(vars,
contribution), y = contribution, gradient = correlation,
label = correlation) + geom_col(aes(fill = correlation)) +
geom_hline(yintercept = mean(plot_data$contribution),
col = "red", lty = "dashed") + scale_fill_gradient2() +
xlab("variable") + coord_flip() + geom_label(color = "black",
fontface = "bold", position = position_dodge(0.5))

}
} else if (!is.na(as.character(mod$Call)[1])) {

if (as.character(mod$Call)[1] == "fa") {
loadings_table = mod$loadings %>% matrix(ncol = ncol(mod$loadings)) %>%
as_tibble() %>% mutate(variable = mod$loadings %>%
rownames()) %>% gather(factor, loading, -variable) %>%
mutate(sign = if_else(loading >= 0, "positive",
"negative"))

if (!is.null(loadings_above)) {
loadings_table[abs(loadings_table[, "loading"]) <
loadings_above, "loading"] = NA
loadings_table = loadings_table[!is.na(loadings_table[,
"loading"]), ]
}

if (!is.null(axes)) {

loadings_table = loadings_table %>% filter(factor ==
paste0("V", axes))
}

plot = loadings_table %>% ggplot() + aes(y = loading %>%
abs(), x = reorder(variable, abs(loading)), fill = loading,
label = round(loading, 2)) + geom_col(position = "dodge") +
scale_fill_gradient2() + coord_flip() + geom_label(color = "black",
fill = "white", fontface = "bold", position = position_dodge(0.5)) +
facet_wrap(~factor) + labs(y = "Loading strength",
x = "Variable")
}
}

return(plot)

```

```
}
```

2.3 A cars beépített adatbázis betöltése

A cars adatbázist fogjuk használni, ami egy beépített adatbázis az R-ben. Az adatbázis 32 különböző automodellről tartalmaz információkat az Motor Trend magazine 1974-es számából. Minden autónak 11 karakterisztikáját tartalmazza az adattábla. Ezek a következők:

- mpg: üzemanyaghatékonyság: hány mérföldet tudunk megtenni az autóval 1 gallon benzinnel
- cyl: hengerek száma
- disp: hengerterfogat
- hp: lóerő
- drat: Rear axle ratio, ennek az üzemanyaghatékonysághoz van köze, minél magasabb ez a szám annál rosszabb az üzemanyaghatékonyság
- wt: súly
- qsec: hány másodperc alatt tesz meg 1/4 mérföldet az autó. a gyorsulással és sebességgel függ össze
- vs: kategorikus változó: a motorblokk alakjáról szól: egyenes alakú (0), V alakú (1)
- am: kategorikus változó: automata váltós autó (0) kézi váltós autó (1).
- gear: sebességek száma
- carb: karburátorok száma

```
data("mtcars")
```

2.4 Az adatsor megtekintése

Vizsgáljuk meg az adatok strukturáját és az alapvető leíró statisztikákat

```
str(mtcars)
```

```
## 'data.frame':   32 obs. of  11 variables:
## $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
## $ cyl : num   6  6  4  6  8  6  8  4  4  6 ...
## $ disp: num  160 160 108 258 360 ...
## $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
## $ drat: num   3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
## $ wt  : num   2.62 2.88 2.32 3.21 3.44 ...
## $ qsec: num  16.5 17 18.6 19.4 17 ...
## $ vs  : num   0  0  1  1  0  1  0  1  1  1 ...
## $ am  : num   1  1  1  0  0  0  0  0  0  0 ...
## $ gear: num   4  4  4  3  3  3  3  4  4  4 ...
## $ carb: num   4  4  1  1  2  1  4  2  2  4 ...
```

```
summary(mtcars)
```

```
##           mpg           cyl           disp           hp
## Min.      :10.40   Min.      :4.000   Min.      : 71.1   Min.      : 52.0
## 1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5
## Median :19.20   Median :6.000   Median :196.3   Median :123.0
## Mean     :20.09   Mean     :6.188   Mean     :230.7   Mean     :146.7
## 3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0
## Max.     :33.90   Max.     :8.000   Max.     :472.0   Max.     :335.0
##           drat           wt           qsec           vs
## Min.      :2.760   Min.      :1.513   Min.      :14.50   Min.      :0.0000
## 1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000
## Median :3.695   Median :3.325   Median :17.71   Median :0.0000
## Mean     :3.597   Mean     :3.217   Mean     :17.85   Mean     :0.4375
```

```
## 3rd Qu.:3.920 3rd Qu.:3.610 3rd Qu.:18.90 3rd Qu.:1.0000
## Max. :4.930 Max. :5.424 Max. :22.90 Max. :1.0000
## am gear carb
## Min. :0.0000 Min. :3.000 Min. :1.000
## 1st Qu.:0.0000 1st Qu.:3.000 1st Qu.:2.000
## Median :0.0000 Median :4.000 Median :2.000
## Mean :0.4062 Mean :3.688 Mean :2.812
## 3rd Qu.:1.0000 3rd Qu.:4.000 3rd Qu.:4.000
## Max. :1.0000 Max. :5.000 Max. :8.000
```

Tegyük fel hogy autót szeretnénk vásárolni és szeretnénk a lehető legjobb döntést hozni azzal kapcsolatban hogy melyik autót válasszuk, de nem értünk különösebben az autókhoz. Amikor megnezzük a Motor Trend magazint, látjuk ezt a 11 karakterisztikát, de nincs idénk minden autót összehasonlítani mind a 11 karakterisztika alapján. Szeretnénk valahogy egyszerűsíteni a helyzetet, és megtudni, mik a legfontosabb alapvető tulajdonságain egy autónak ami alapján a döntésünket meghozhatjuk.

Nezzük meg az adatok korrelációs matrixát, hogy jobb képet kapjunk arról, mely változók függenek össze egymással és milyen szorosan.

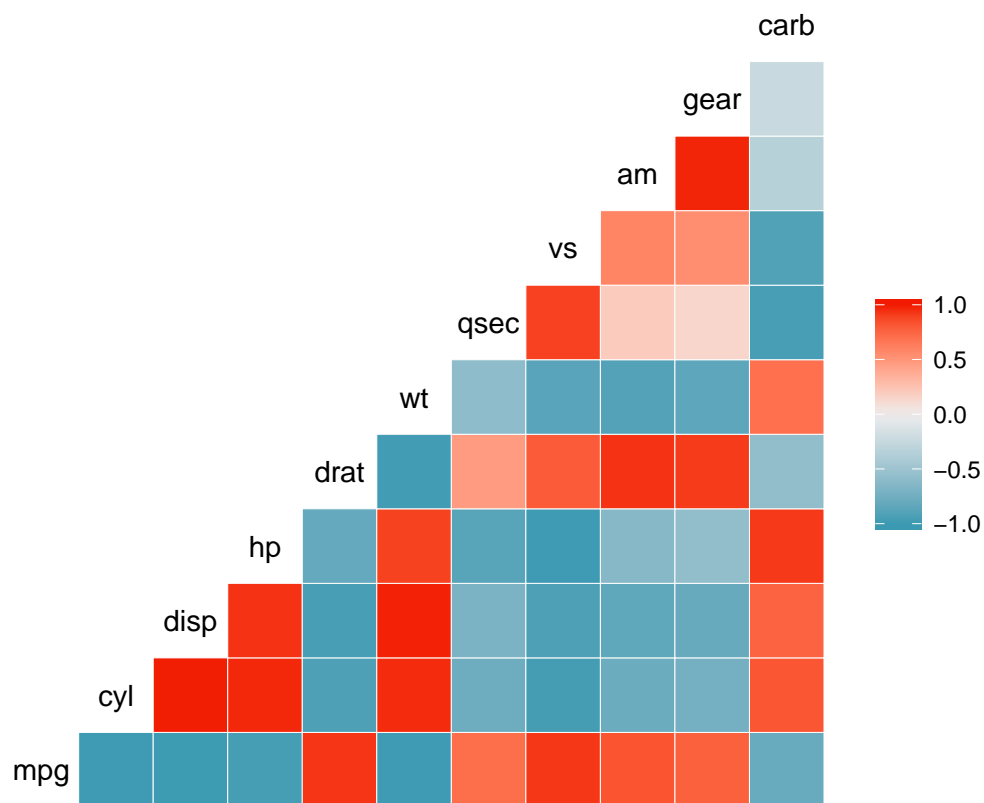
A korrelációs matrixot többfajta módon is vizualizálhatjuk. Az eredmény alapján látható hogy vannak korrelációs “klaszterek”, vagyis olyan változók amik csoportosan összefüggenek egymással, de a kép egyelőre túl bonyolult, mert túl sok a változó.

Az alábbi példában használt `network_plot()` funció neha nem ranzolja ki a vonalakat. Sajnos ennek nem jöttem rá a megoldására. Ezt mások is tapasztaltak, itt lehet követni ezt a hibajelentést: <https://github.com/tidymodels/corrr/issues/63>

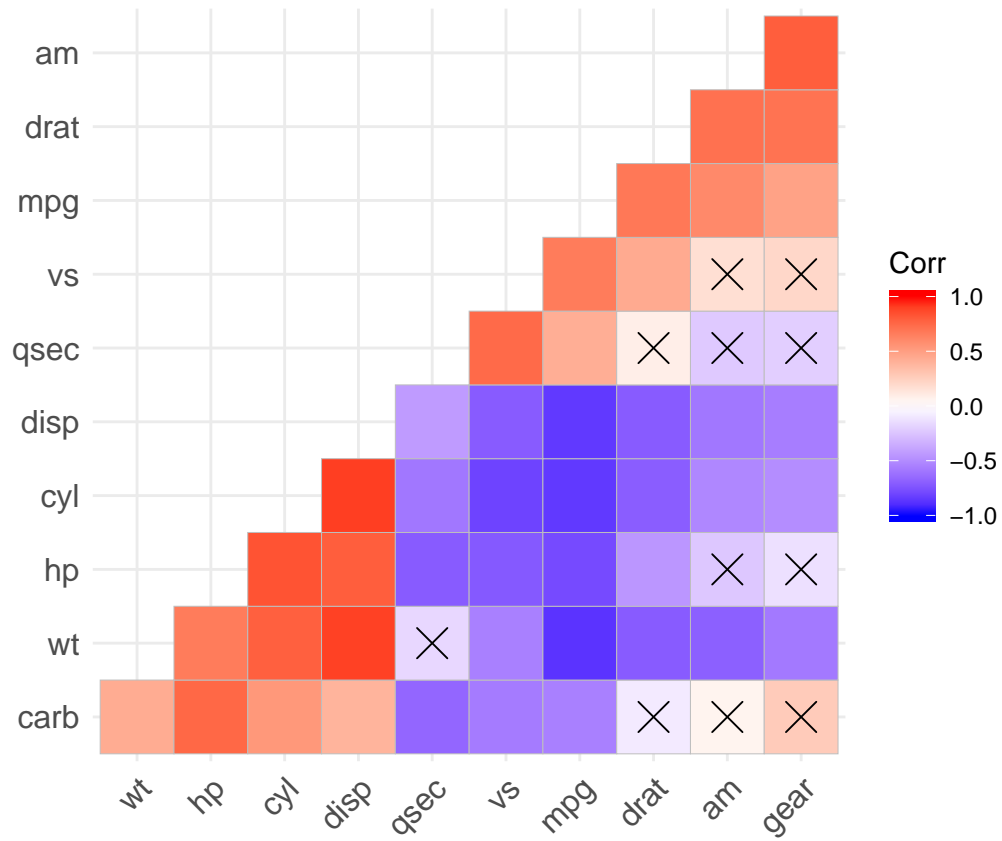
```
cor = mtcars %>% cor()
cor
```

```
## mpg cyl disp hp drat wt
## mpg 1.0000000 -0.8521620 -0.8475514 -0.7761684 0.68117191 -0.8676594
## cyl -0.8521620 1.0000000 0.9020329 0.8324475 -0.69993811 0.7824958
## disp -0.8475514 0.9020329 1.0000000 0.7909486 -0.71021393 0.8879799
## hp -0.7761684 0.8324475 0.7909486 1.0000000 -0.44875912 0.6587479
## drat 0.6811719 -0.6999381 -0.7102139 -0.4487591 1.00000000 -0.7124406
## wt -0.8676594 0.7824958 0.8879799 0.6587479 -0.71244065 1.0000000
## qsec 0.4186840 -0.5912421 -0.4336979 -0.7082234 0.09120476 -0.1747159
## vs 0.6640389 -0.8108118 -0.7104159 -0.7230967 0.44027846 -0.5549157
## am 0.5998324 -0.5226070 -0.5912270 -0.2432043 0.71271113 -0.6924953
## gear 0.4802848 -0.4926866 -0.5555692 -0.1257043 0.69961013 -0.5832870
## carb -0.5509251 0.5269883 0.3949769 0.7498125 -0.09078980 0.4276059
## qsec vs am gear carb
## mpg 0.41868403 0.6640389 0.59983243 0.4802848 -0.55092507
## cyl -0.59124207 -0.8108118 -0.52260705 -0.4926866 0.52698829
## disp -0.43369788 -0.7104159 -0.59122704 -0.5555692 0.39497686
## hp -0.70822339 -0.7230967 -0.24320426 -0.1257043 0.74981247
## drat 0.09120476 0.4402785 0.71271113 0.6996101 -0.09078980
## wt -0.17471588 -0.5549157 -0.69249526 -0.5832870 0.42760594
## qsec 1.00000000 0.7445354 -0.22986086 -0.2126822 -0.65624923
## vs 0.74453544 1.0000000 0.16834512 0.2060233 -0.56960714
## am -0.22986086 0.1683451 1.00000000 0.7940588 0.05753435
## gear -0.21268223 0.2060233 0.79405876 1.0000000 0.27407284
## carb -0.65624923 -0.5696071 0.05753435 0.2740728 1.00000000
```

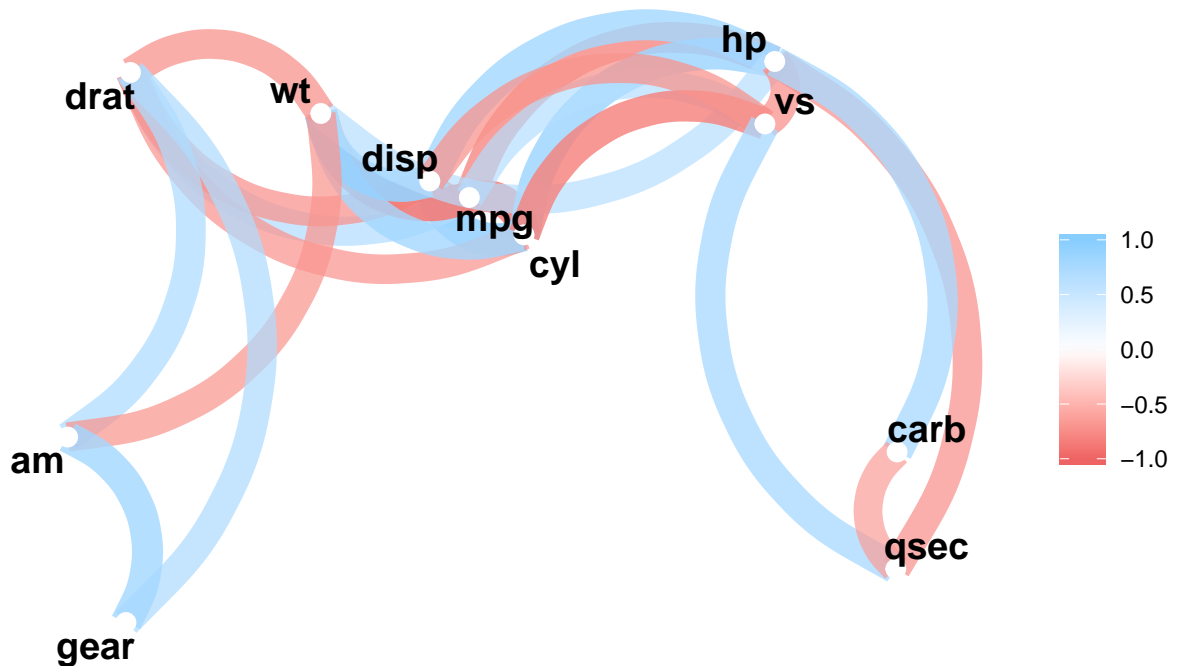
```
ggcorr(cor)
```



```
ggcorrplot(cor(mtcars), p.mat = cor_pmat(mtcars), hc.order = TRUE,
  type = "lower")
```



```
cor %>% network_plot(min_cor = 0.6)
```



3 Fokomponenselemzés

3.1 A fokomponenselemzés modell megepítése

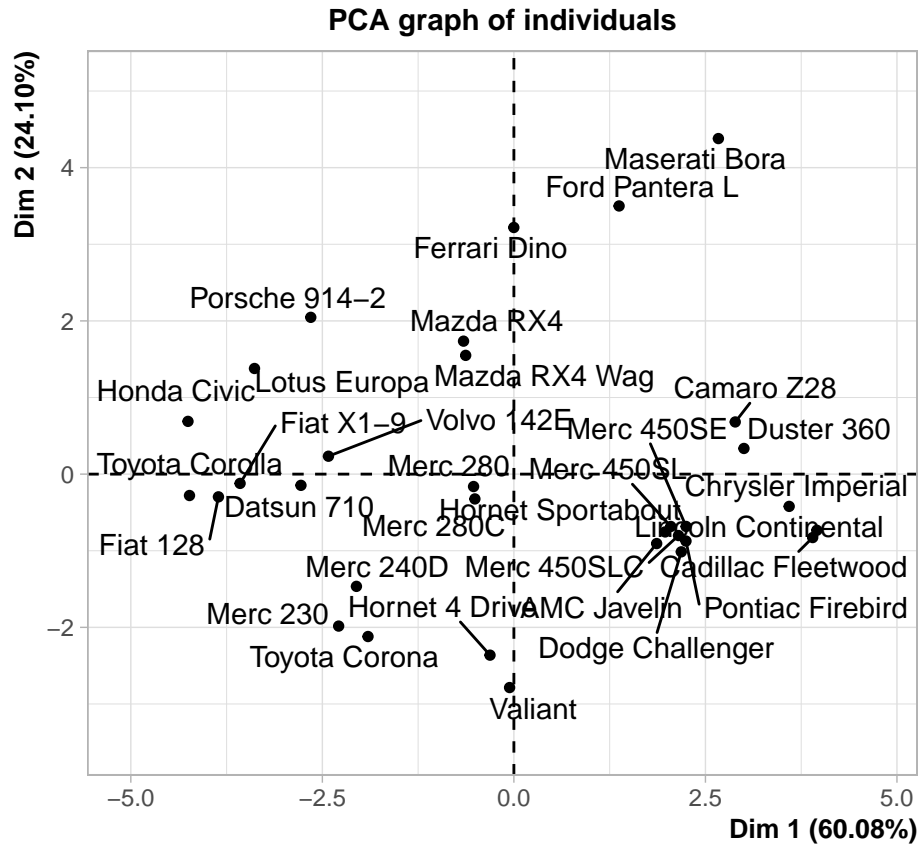
Néhány változó összefüggését könnyen átlathatóvá tehetjük vizualizáción keresztül. Azonban ha sok változóval van dolgunk, a vizualizáció és egyéb korábban tanult feltárási adatkezelési technikák kudarcot vallhatnak egyszerűen azért mert túl sok az információ amit nehéz átlátni és vizualizálni. Ebben a helyzetben segíthet a fokomponenselemzés, amit arra használunk hogy lecsökkentsük a változók számát amivel dolgoznunk kell, úgy, hogy közben a lehető legtöbb információt tartunk meg az adatok variabilitásáról.

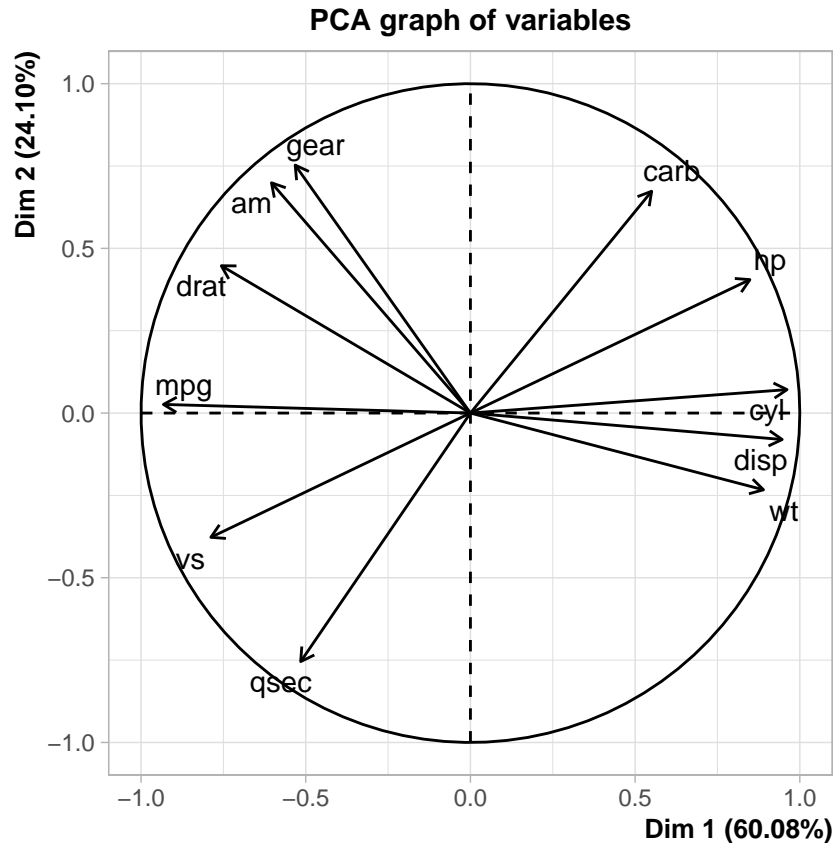
A fokomponenselemzést a `PCA()` (principal component analysis) funkcióval tudjuk elvégezni a `FactoMineR` csomagból. Az elemzés eredményét egy `pca_mod` modell objektumba mentettem el. A egyből kiad két ábrát a két legfontosabb fokomponensről (dimenzióról) amik a leghatékonyabban írják le az adatokat.

Az egyik ábrán az látszik, hogy az egyes megfigyelések (ebben az esetben az egyes auto-modellek) hol helyezkednek el a két dimenzió mentén. A második ábra pedig arról szól, hogy a dimenziók milyen korrelációt mutatnak az eredeti változokkal. A szaggatott vonalak mutatják a fokomponenseket. A nyílak minél közelebb fekszenek a szaggatott vonalhoz, a változó annál inkább együttjár az adott dimenzióval a másik dimenzióval szemben.

Peladatul a `cyl` és az `mpg` változókat sokkal jobban leírja a Dim1 mint a Dim2. (a nyíl iránya alapján megállapítható hogy az `mpg` negatívan, a `cyl` pozitívan korrelál a Dim1-el.) Ezzel szemben az `carb` változó nyílja a két dimenzió között helyezkedik el, ami azt jelenti hogy vagy midkettővel nagyjából azonos mértékben korrelálnak (ez lehet nagyon kicsi, vagy akár nagyon nagy korreláció is).

```
pca_mod <- PCA(mtcars)
```



Arra oda kell figyelnünk hogy kategorikus változók ne kerüljenek a főkomponenselemzés változói közé. a vs és am változók kategorikus változók (csak 0 és 1 értéket vehetnek fel), és a cyl változó is tekinthető kategorikusnak, hiszen csak három értéket vesz fel az adattáblánkban: 4, 6, 8.

A `PCA()` funkcióban lehetőségek van arra hogy meghatározzunk olyan változókat az adatbázisban, amiket nem szeretnénk beépíteni a PCA modellbe. Azokat a folytonos változókat, amiket nem szeretnénk figyelembe venni a PCA során, a `quant.sup` parameterben kell megadnunk, azokat pedig amik kategorikusak a `quali.sup` parameterben. Itt az adott változó oszlopszámát kell megadnunk, nem pedig a nevet, így ezt először ki kell keresnünk. Ezt megtehetjük a `which(names(mtcars) == "változó neve")` funkcióval.

Az alábbi példában a `drat` folytonos, és a `cyl`, `vs`, és az `am` kategorikus változókat kiemeljük a modellből, így azok nincsenek figyelembe véve a `pca_mod3` főkomponenseinek meghatározása során, de az abszolút értékek meg szerepelnek. Ebben az esetben természetesen a modell újra illesztésre kerül, és a számszerű értékek megváltoznak a korábbi futtatáshoz képest amikor ezek a változók meg szerepeltek a modellben.

```
which(names(mtcars) == "drat")
```

```
## [1] 5
```

```
which(names(mtcars) == "cyl")
```

```
## [1] 2
```

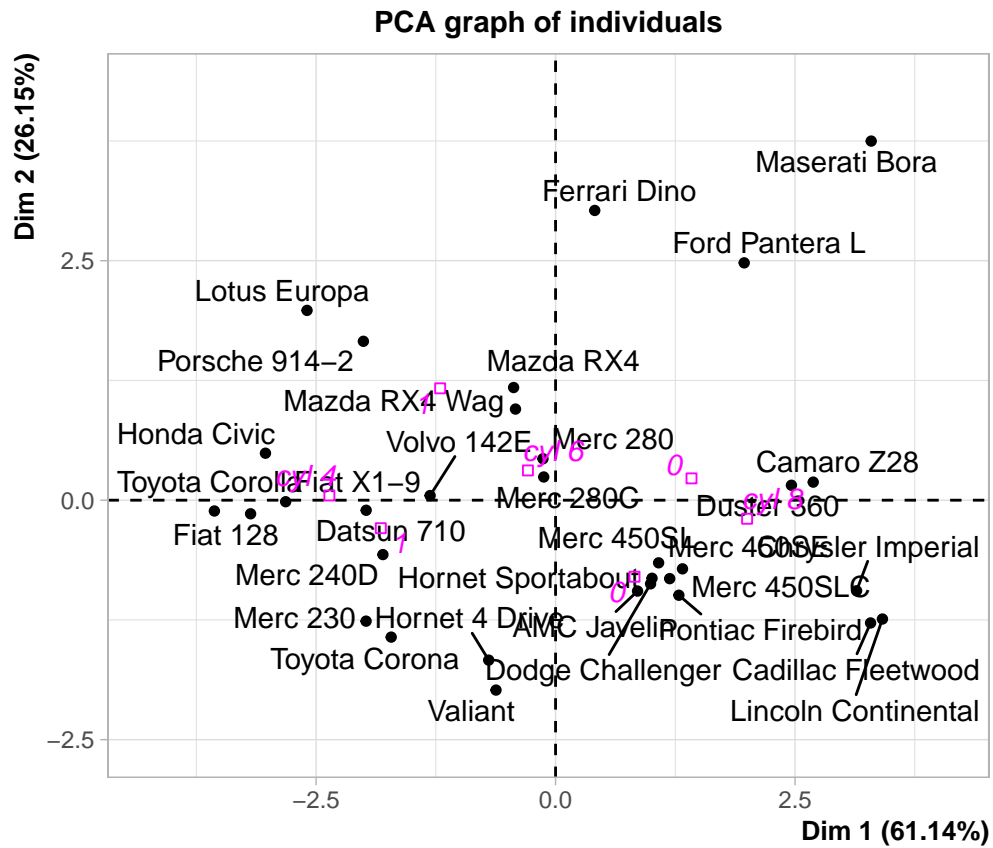
```
which(names(mtcars) == "vs")
```

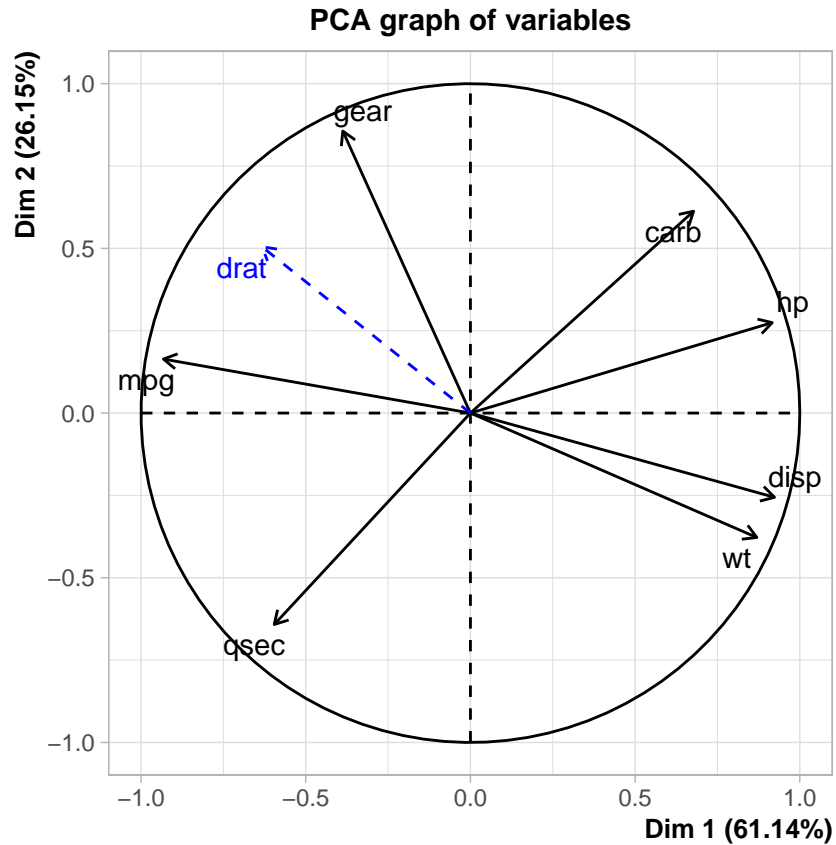
```
## [1] 8
```

```
which(names(mtcars) == "am")
```

```
## [1] 9
```

```
pca_mod2 <- PCA(mtcars, quanti.sup = 5, quali.sup = c(2, 8, 9))
```





```
summary(pca_mod2)
```

```
##
## Call:
## PCA(X = mtcars, quanti.sup = 5, quali.sup = c(2, 8, 9))
##
## Eigenvalues
##           Dim.1  Dim.2  Dim.3  Dim.4  Dim.5  Dim.6  Dim.7
## Variance      4.280   1.831   0.458   0.184   0.120   0.104   0.024
## % of var.      61.138  26.151   6.550   2.624   1.713   1.488   0.337
## Cumulative % of var. 61.138  87.289  93.839  96.462  98.175  99.663 100.000
##
## Individuals (the 10 first)
##           Dist  Dim.1  ctr  cos2  Dim.2  ctr  cos2
## Mazda RX4      | 1.551 | -0.438 0.140 0.080 | 1.176 2.361 0.575 |
## Mazda RX4 Wag  | 1.322 | -0.418 0.128 0.100 | 0.951 1.544 0.518 |
## Datsun 710      | 2.096 | -1.977 2.854 0.890 | -0.105 0.019 0.003 |
## Hornet 4 Drive  | 1.846 | -0.698 0.356 0.143 | -1.670 4.759 0.818 |
## Hornet Sportabout | 1.669 | 1.007 0.741 0.364 | -0.815 1.134 0.238 |
## Valiant         | 2.139 | -0.622 0.282 0.084 | -1.982 6.709 0.859 |
## Duster 360      | 2.663 | 2.462 4.428 0.855 | 0.156 0.041 0.003 |
## Merc 240D       | 2.126 | -1.802 2.372 0.719 | -0.568 0.551 0.071 |
## Merc 230        | 3.168 | -1.978 2.857 0.390 | -1.263 2.723 0.159 |
## Merc 280        | 1.130 | -0.131 0.013 0.014 | 0.433 0.320 0.147 |
##
##           Dim.3  ctr  cos2
```

```

## Mazda RX4          -0.116  0.092  0.006 |
## Mazda RX4 Wag      0.196  0.262  0.022 |
## Datsun 710         -0.276  0.520  0.017 |
## Hornet 4 Drive     -0.276  0.519  0.022 |
## Hornet Sportabout -0.995  6.746  0.355 |
## Valiant            0.256  0.447  0.014 |
## Duster 360        -0.759  3.930  0.081 |
## Merc 240D          0.839  4.802  0.156 |
## Merc 230           1.944 25.755  0.376 |
## Merc 280           0.917  5.729  0.659 |
##
## Variables
##              Dim.1   ctr   cos2   Dim.2   ctr   cos2   Dim.3   ctr
## mpg          | -0.932 20.294 0.869 | 0.164 1.469 0.027 | -0.095 1.971
## disp         | 0.923 19.913 0.852 | -0.256 3.572 0.065 | -0.136 4.047
## hp           | 0.916 19.612 0.839 | 0.275 4.119 0.075 | -0.058 0.740
## wt           | 0.869 17.664 0.756 | -0.378 7.792 0.143 | 0.249 13.472
## qsec         | -0.596 8.289 0.355 | -0.642 22.505 0.412 | 0.458 45.663
## gear         | -0.387 3.506 0.150 | 0.856 40.036 0.733 | 0.208 9.465
## carb         | 0.677 10.721 0.459 | 0.613 20.508 0.375 | 0.336 24.641
##              cos2
## mpg          0.009 |
## disp         0.019 |
## hp           0.003 |
## wt           0.062 |
## qsec         0.209 |
## gear         0.043 |
## carb         0.113 |
##
## Supplementary continuous variable
##              Dim.1   cos2   Dim.2   cos2   Dim.3   cos2
## drat          | -0.633 0.400 | 0.505 0.255 | 0.083 0.007 |
##
## Supplementary categories
##              Dist   Dim.1   cos2 v.test   Dim.2   cos2 v.test
## cyl 4          | 2.367 | -2.361 0.995 -4.600 | 0.049 0.000 0.147 |
## cyl 6          | 0.711 | -0.289 0.165 -0.411 | 0.311 0.191 0.676 |
## cyl 8          | 2.025 | 2.000 0.975 4.746 | -0.194 0.009 -0.704 |
## 0              | 1.449 | 1.419 0.960 4.331 | 0.227 0.025 1.061 |
## 1              | 1.862 | -1.824 0.960 -4.331 | -0.292 0.025 -1.061 |
## 0              | 1.151 | 0.824 0.513 2.682 | -0.799 0.482 -3.975 |
## 1              | 1.682 | -1.205 0.513 -2.682 | 1.168 0.482 3.975 |
##              Dim.3   cos2 v.test
## cyl 4          0.049 0.000 0.289 |
## cyl 6          0.393 0.305 1.710 |
## cyl 8         -0.235 0.013 -1.701 |
## 0             -0.169 0.014 -1.579 |
## 1              0.218 0.014 1.579 |
## 0              0.067 0.003 0.666 |
## 1             -0.098 0.003 -0.666 |

```

3.2 Hany uj dimenziot generaljunk?

A fokkomponenselemzes egy dmienzioredukcios technika, vagyis celunk hogy kevesebb dimenzionk legyen az elemzes vegere, mint ahany változóval kezdtük az elemzést. Viszont hogyha ranezunk a model summary-ra, lathatjuk hogy a PCA funkcio alapertelmezett modon pontosan annyi dimenziot generalt mint amennyi változónk volt. Meg kell mondanunk a PCA funkcionak, hany dimenziot akarunk kinyerni. De hogyan tudjuk eldönteni, mennyi az idealis szamu dimenzio?

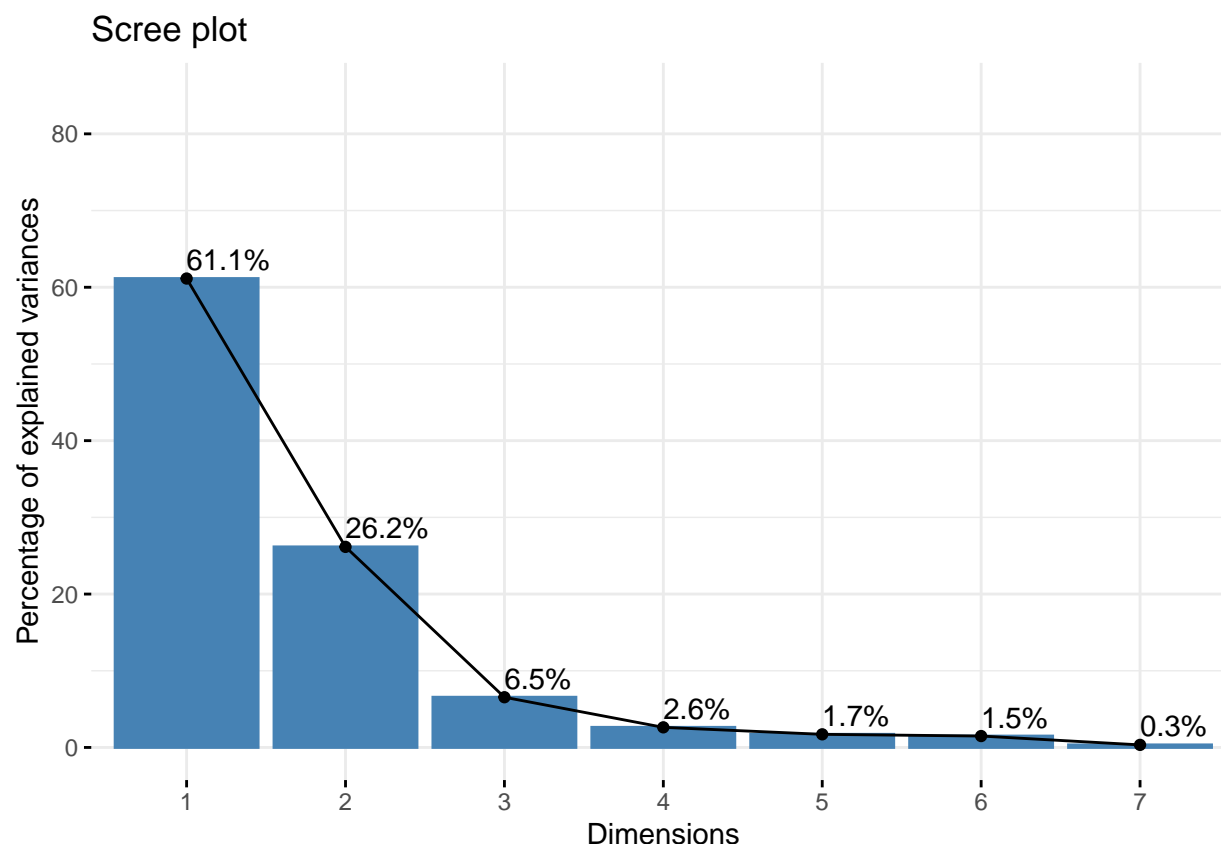
Erre szamos modszer letezik.

1. Scree test

A legismertebb talan a scree-test, ami a megmagyarazott varianciaaranyt abrazolo abra alapján vegezhető el. Ehhez eloszor a `fviz_screplot()` funkcióval abrazolnunk kell az egyes fokkomponensek által megmagyarazott variancia merteket, majd az abra alapján meg kell allapitanunk, hol van a “tores” a scree-plotban, vagyis hol talalhato az a pont, ami utan mar ellaposodik a megmagyarazott varianciaaranyt abrazolo gorbe. A torespont elotti dimenzional kell hogy megalljon a dimenzio-extrakcio, vagyis annal a dimenzional, ami meg szignifikansan tobb varianciat képes megmagyarazni, mint a kesobb kinyerd dimenziok. Ezt a megallasi szabalyt ugy is nevezik hogy a “konyok kriterium”, mivel a scree plot egy konyokra emlekeztet, es mi a konyokpontot keressuk a gorbeben.

Ezen az abran ugy tunik, hogy a masodik dimenzio utan mar nem erdemes tovabbmennunk, hiszen a harmadik dimenzional megtorik a gorbe es onnantol mar nagyon alacsony a megmagyarazott varianciaarany. Vagyis az idealis dimenzioszam ezen az adaton 2.

```
fviz_screplot(pca_mod2, addlabels = TRUE, ylim = c(0, 85))
```



The Kaiser-Guttman szabaly

Egy masik jól ismert kriterium, hogy azokat a dimenziokat kell megtartanunk, amelyeknek az eigenvalue

erteke 1-nel magasabb. Ez azért van, mert az 1-nel alacsonyabb eigenvalue azt jelenti, hogy a dimenzio kevesebb varianciát magyaráz meg mint az eredeti változók átlagosan. A főkomponens elemzés lényege hogy hasznos összefoglaló változókat generáljunk amik több változó információját tartalmazzak összevonva. Azt pedig nem szeretnénk hogy meg az eredeti változóinknál is haszontalanabb változókat generáljunk, ezért az átlagos változóinknál kisebb varianciát megmagyarázó dimenziókat elutasítjuk.

A példánkban ez az elemzés is azt sugallja, hogy két dimenziót tartunk meg, hiszen a harmadik dimenzióhoz tartozó eigenvalue már 1-nél kisebb.

```
get_eigenvalue(pca_mod2)
```

```
##          eigenvalue variance.percent cumulative.variance.percent
## Dim.1 4.27967149      61.1381642      61.13816
## Dim.2 1.83055831      26.1508331      87.28900
## Dim.3 0.45848582       6.5497975     93.83879
## Dim.4 0.18365828       2.6236897     96.46248
## Dim.5 0.11988871       1.7126958     98.17518
## Dim.6 0.10416965       1.4881379     99.66332
## Dim.7 0.02356773       0.3366819    100.00000
```

Parallel elemzés

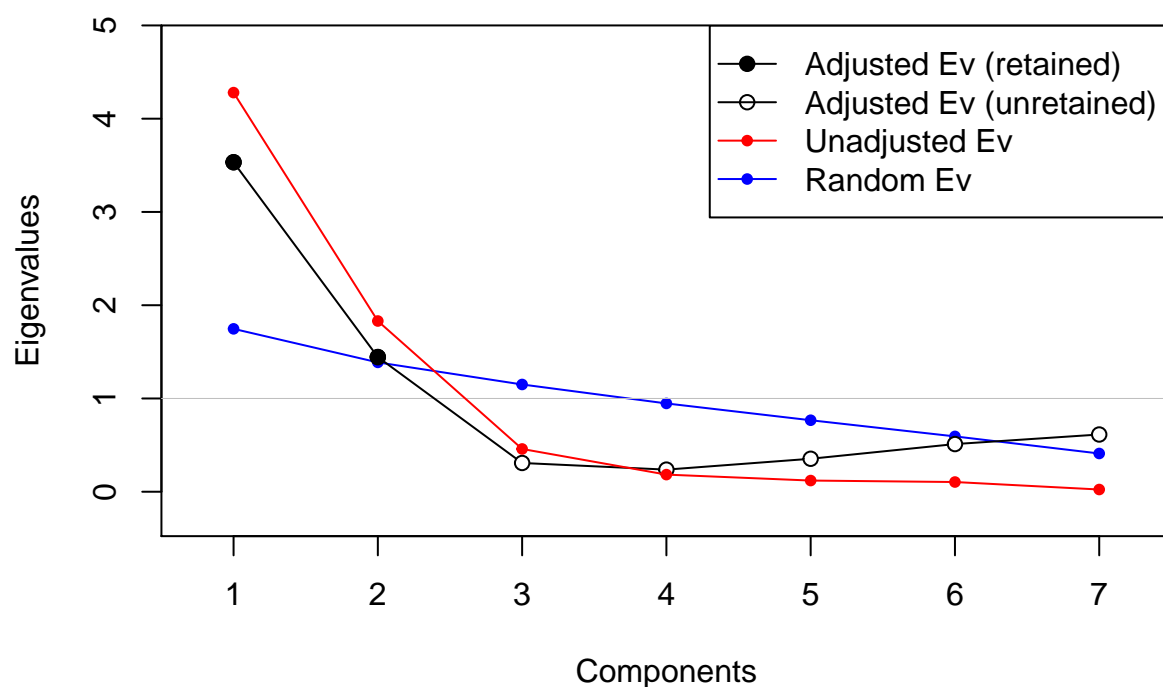
A harmadik (és egyben jelenleg a legelfogadottabb) technika a parallel elemzés technika. Ennek a lényege hogy az eredeti adattáblánkhoz hasonló karakterisztikákkal rendelkező adatokat generálunk véletlenszerűen, de úgy, hogy abban a változók ne korreláljanak egymással. Ezt nagyon sokszor megismételjük, és ez alapján a nagy mennyiségű random minta alapján kiszámoljuk, mi a véletlenszerűen várható eigenvalue mintázat. Ez egyfajta “null modelként” funkcionál, amihez hasonlíthatjuk a saját adatainkon kapott eigenvalue-kat. Azokat a dimenziókat tartjuk meg, amiknek az eigenvalue-ja magasabb mint a random mintákban az adott dimenzióhoz tartozó null-eigenvalue.

Ezt a parallel elemzést bevezethetjük el a paran() funkcióval a paran package-ból. Ez a funkció a null eigenvalue görbe vizualizálására is képes a graph = TRUE prameter beállításával, melyet összehasonlíthatunk az adatainkban kapott eigenvalue-val. Az output objektum \$Retained komponense megmutatja, az elemzés hány dimenzio megtartását javasolja.

```
mtcars_pca_ret = paran(mtcars[, -c(2, 5, 8, 9)], graph = TRUE)
```

```
##
## Using eigendecomposition of correlation matrix.
## Computing: 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
##
##
## Results of Horn's Parallel Analysis for component retention
## 210 iterations, using the mean estimate
##
## -----
## Component    Adjusted    Unadjusted    Estimated
##              Eigenvalue Eigenvalue    Bias
## -----
## 1             3.533042    4.279671    0.746629
## 2             1.443866    1.830558    0.386692
## -----
##
## Adjusted eigenvalues > 1 indicate dimensions to retain.
## (2 components retained)
```

Parallel Analysis

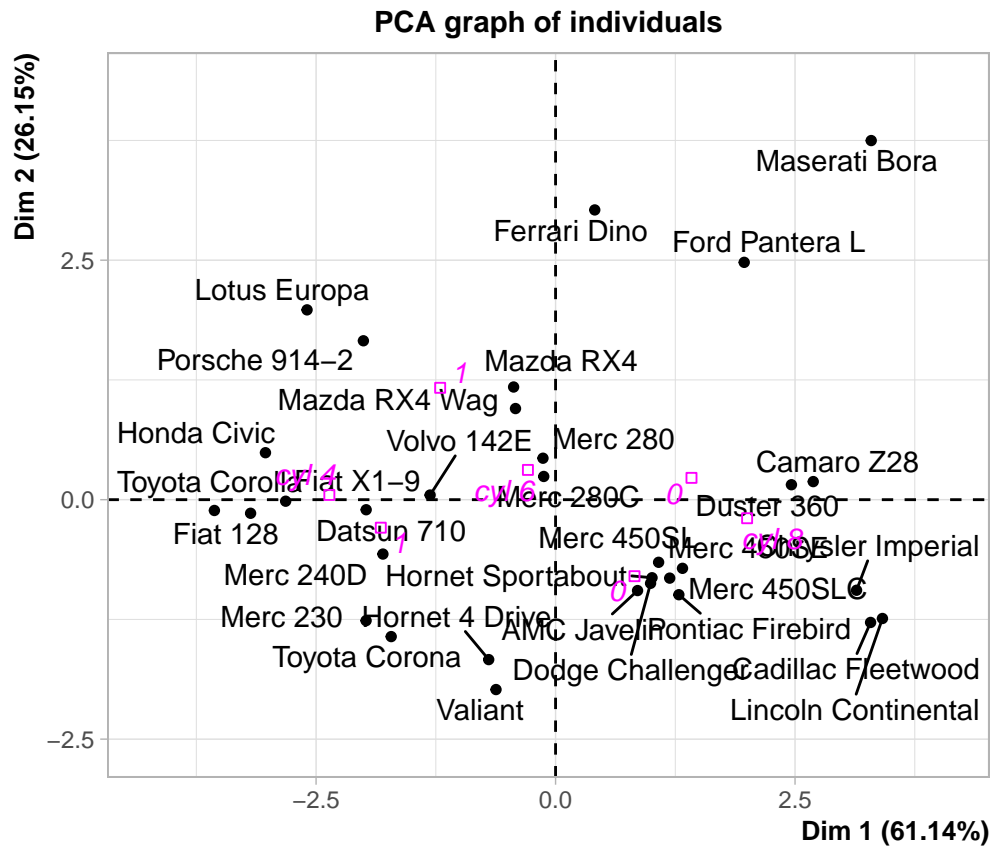


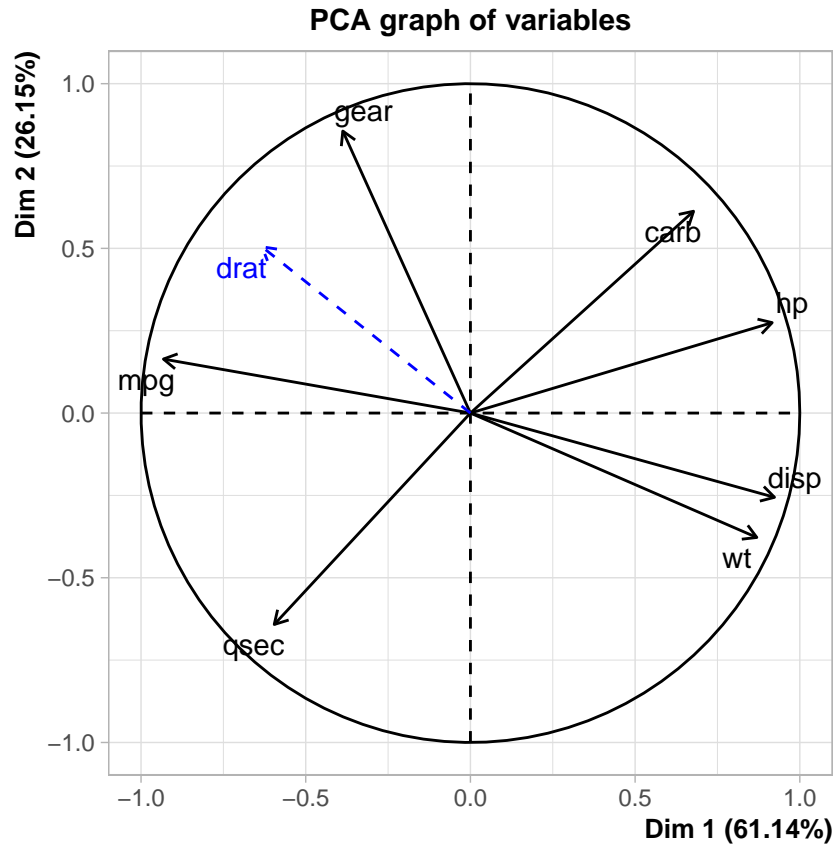
```
mtcars_pca_ret$Retained
```

```
## [1] 2
```

Amint meghatároztuk az ideális dimenziók számát, újra lefuttathatjuk az elemzésünket, de ezúttal már specifikálva, mennyi dimenziót szeretnénk, a `npc` parameter beállításával.

```
pca_mod3 <- PCA(mtcars, ncp = 2, quanti.sup = 5, quali.sup = c(2,  
8, 9))
```



```
summary(pca_mod3)
```

```
##
## Call:
## PCA(X = mtcars, ncp = 2, quanti.sup = 5, quali.sup = c(2, 8,
##      9))
##
##
## Eigenvalues
##           Dim.1   Dim.2   Dim.3   Dim.4   Dim.5   Dim.6   Dim.7
## Variance      4.280   1.831   0.458   0.184   0.120   0.104   0.024
## % of var.     61.138  26.151   6.550   2.624   1.713   1.488   0.337
## Cumulative % of var. 61.138  87.289  93.839  96.462  98.175  99.663 100.000
##
## Individuals (the 10 first)
##           Dist   Dim.1   ctr   cos2   Dim.2   ctr   cos2
## Mazda RX4      | 1.551 | -0.438 0.140 0.080 | 1.176 2.361 0.575 |
## Mazda RX4 Wag  | 1.322 | -0.418 0.128 0.100 | 0.951 1.544 0.518 |
## Datsun 710      | 2.096 | -1.977 2.854 0.890 | -0.105 0.019 0.003 |
## Hornet 4 Drive  | 1.846 | -0.698 0.356 0.143 | -1.670 4.759 0.818 |
## Hornet Sportabout | 1.669 | 1.007 0.741 0.364 | -0.815 1.134 0.238 |
## Valiant         | 2.139 | -0.622 0.282 0.084 | -1.982 6.709 0.859 |
## Duster 360      | 2.663 | 2.462 4.428 0.855 | 0.156 0.041 0.003 |
## Merc 240D       | 2.126 | -1.802 2.372 0.719 | -0.568 0.551 0.071 |
## Merc 230        | 3.168 | -1.978 2.857 0.390 | -1.263 2.723 0.159 |
## Merc 280        | 1.130 | -0.131 0.013 0.014 | 0.433 0.320 0.147 |
```

```
##
## Variables
##           Dim.1   ctr   cos2   Dim.2   ctr   cos2
## mpg          | -0.932 20.294 0.869 | 0.164 1.469 0.027 |
## disp          | 0.923 19.913 0.852 | -0.256 3.572 0.065 |
## hp            | 0.916 19.612 0.839 | 0.275 4.119 0.075 |
## wt            | 0.869 17.664 0.756 | -0.378 7.792 0.143 |
## qsec          | -0.596 8.289 0.355 | -0.642 22.505 0.412 |
## gear          | -0.387 3.506 0.150 | 0.856 40.036 0.733 |
## carb          | 0.677 10.721 0.459 | 0.613 20.508 0.375 |
##
## Supplementary continuous variable
##           Dim.1   cos2   Dim.2   cos2
## drat          | -0.633 0.400 | 0.505 0.255 |
##
## Supplementary categories
##           Dist   Dim.1   cos2 v.test   Dim.2   cos2 v.test
## cyl 4          | 2.367 | -2.361 0.995 -4.600 | 0.049 0.000 0.147 |
## cyl 6          | 0.711 | -0.289 0.165 -0.411 | 0.311 0.191 0.676 |
## cyl 8          | 2.025 | 2.000 0.975 4.746 | -0.194 0.009 -0.704 |
## 0              | 1.449 | 1.419 0.960 4.331 | 0.227 0.025 1.061 |
## 1              | 1.862 | -1.824 0.960 -4.331 | -0.292 0.025 -1.061 |
## 0              | 1.151 | 0.824 0.513 2.682 | -0.799 0.482 -3.975 |
## 1              | 1.682 | -1.205 0.513 -2.682 | 1.168 0.482 3.975 |
```

3.3 Fokomponenselemzés eredményeinek értelmezése

3.3.1 a PCA modell objektum részei

A modell összefoglalóbol (model summary) további hasznos információk olvashatók ki. Az Eigenvalues részben megtudhatjuk hogy az egyes dimenziók az adatok teljes varianciajának hány százalékát magyarázzák meg (% of var), és hogy a legfontosabbtól a legalacsonyabbig egyesevel összevonva mekkora a több dimenzio által megmagyarázott összesített varianciaarány (Cumulative % of var). Vagyis a Dim.3-hoz tartozó % of variance érték (6.55) azt mutatja, hogy a harmadikként kinyert dimenzio az adatok varianciajának 65%-át tudja megmagyarázni onmagában. Az Dim.3-hoz tartozó dim Cumulative % of var érték (93.84 pedig azt mutatja, hogy a Dimenzio 3 a Dim.1 és Dim.2-vel együtt közösen az adatok varianciajának 93.84%-át képesek megmagyarázni. Ha csak az eigenvalue-t és a megmagyarázott varianciaarányokat tartalmazó táblázat érdekli minket, ezt kinyerhetjük úgy hogy csak a `pca_mod3$eig` komponenst listázzuk ki.

A model summary arrol is tartalmaz információt a Variables részben, hogy az egyes változók hogyan korrelálnak az egyes új dimenziókkal (a Dim.1, Dim.2, Dim.2 ... oszlopokban), és hogy mekkora a hozzájárulásuk az adott változó által megmagyarázott varianciahoz (a ctr oszlopban). Ez egy nagyon fontos táblázat, mert innen tudjuk leolvasni (az abrak mellett) hogy az egyes változókat mely dimenziók (faktorok) írják le leginkább. Bővebb információt találunk ha kilistázzuk a `pca_mod3$var` komponenst.

```
# Get the summary the outputs.
```

```
summary(pca_mod3)
```

```
##
## Call:
## PCA(X = mtcars, ncp = 2, quanti.sup = 5, quali.sup = c(2, 8,
##           9))
##
##
## Eigenvalues
##           Dim.1   Dim.2   Dim.3   Dim.4   Dim.5   Dim.6   Dim.7
```

```

## Variance          4.280   1.831   0.458   0.184   0.120   0.104   0.024
## % of var.        61.138  26.151   6.550   2.624   1.713   1.488   0.337
## Cumulative % of var. 61.138  87.289  93.839  96.462  98.175  99.663 100.000
##
## Individuals (the 10 first)
##          Dist   Dim.1   ctr   cos2   Dim.2   ctr   cos2
## Mazda RX4      | 1.551 | -0.438 0.140 0.080 | 1.176 2.361 0.575 |
## Mazda RX4 Wag  | 1.322 | -0.418 0.128 0.100 | 0.951 1.544 0.518 |
## Datsun 710      | 2.096 | -1.977 2.854 0.890 | -0.105 0.019 0.003 |
## Hornet 4 Drive  | 1.846 | -0.698 0.356 0.143 | -1.670 4.759 0.818 |
## Hornet Sportabout | 1.669 | 1.007 0.741 0.364 | -0.815 1.134 0.238 |
## Valiant         | 2.139 | -0.622 0.282 0.084 | -1.982 6.709 0.859 |
## Duster 360      | 2.663 | 2.462 4.428 0.855 | 0.156 0.041 0.003 |
## Merc 240D       | 2.126 | -1.802 2.372 0.719 | -0.568 0.551 0.071 |
## Merc 230        | 3.168 | -1.978 2.857 0.390 | -1.263 2.723 0.159 |
## Merc 280        | 1.130 | -0.131 0.013 0.014 | 0.433 0.320 0.147 |
##
## Variables
##          Dim.1   ctr   cos2   Dim.2   ctr   cos2
## mpg          | -0.932 20.294 0.869 | 0.164 1.469 0.027 |
## disp         | 0.923 19.913 0.852 | -0.256 3.572 0.065 |
## hp           | 0.916 19.612 0.839 | 0.275 4.119 0.075 |
## wt           | 0.869 17.664 0.756 | -0.378 7.792 0.143 |
## qsec         | -0.596 8.289 0.355 | -0.642 22.505 0.412 |
## gear         | -0.387 3.506 0.150 | 0.856 40.036 0.733 |
## carb         | 0.677 10.721 0.459 | 0.613 20.508 0.375 |
##
## Supplementary continuous variable
##          Dim.1   cos2   Dim.2   cos2
## drat         | -0.633 0.400 | 0.505 0.255 |
##
## Supplementary categories
##          Dist   Dim.1   cos2 v.test   Dim.2   cos2 v.test
## cyl 4        | 2.367 | -2.361 0.995 -4.600 | 0.049 0.000 0.147 |
## cyl 6        | 0.711 | -0.289 0.165 -0.411 | 0.311 0.191 0.676 |
## cyl 8        | 2.025 | 2.000 0.975 4.746 | -0.194 0.009 -0.704 |
## 0            | 1.449 | 1.419 0.960 4.331 | 0.227 0.025 1.061 |
## 1            | 1.862 | -1.824 0.960 -4.331 | -0.292 0.025 -1.061 |
## 0            | 1.151 | 0.824 0.513 2.682 | -0.799 0.482 -3.975 |
## 1            | 1.682 | -1.205 0.513 -2.682 | 1.168 0.482 3.975 |

```

```
pca_mod3$eig
```

```

##          eigenvalue percentage of variance cumulative percentage of variance
## comp 1 4.27967149          61.1381642          61.13816
## comp 2 1.83055831          26.1508331          87.28900
## comp 3 0.45848582          6.5497975          93.83879
## comp 4 0.18365828          2.6236897          96.46248
## comp 5 0.11988871          1.7126958          98.17518
## comp 6 0.10416965          1.4881379          99.66332
## comp 7 0.02356773          0.3366819          100.00000

```

```
pca_mod3$var
```

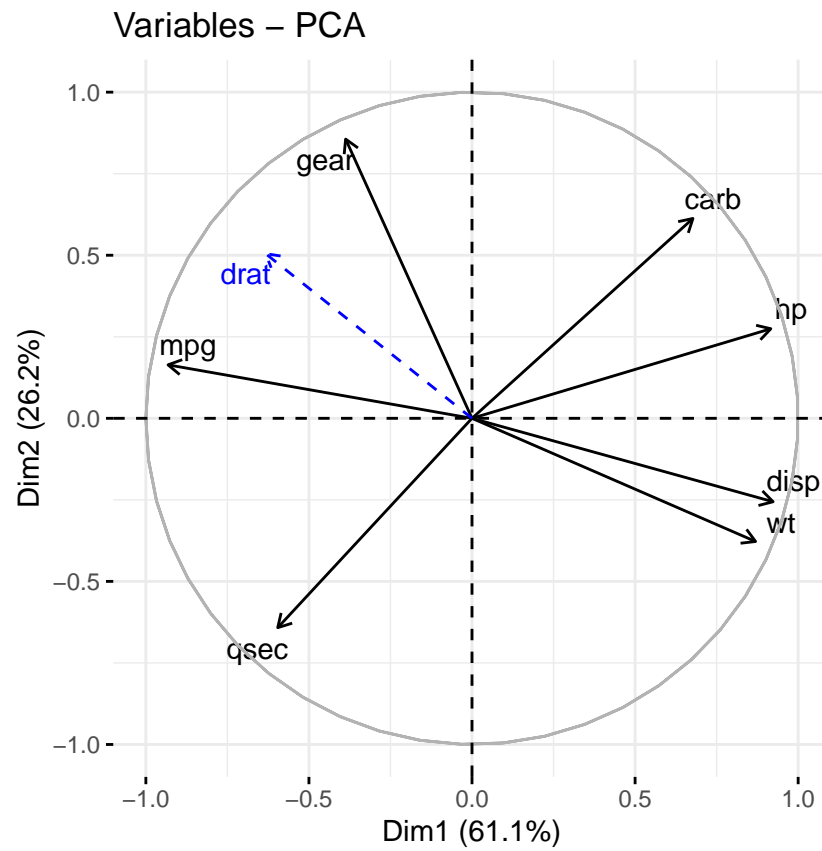
```
## $coord
```

```
##          Dim.1      Dim.2
## mpg  -0.9319530  0.1639662
## disp  0.9231535 -0.2557244
## hp    0.9161546  0.2745792
## wt    0.8694559 -0.3776643
## qsec -0.5956200 -0.6418408
## gear -0.3873752  0.8560856
## carb  0.6773530  0.6127089
##
## $cor
##          Dim.1      Dim.2
## mpg  -0.9319530  0.1639662
## disp  0.9231535 -0.2557244
## hp    0.9161546  0.2745792
## wt    0.8694559 -0.3776643
## qsec -0.5956200 -0.6418408
## gear -0.3873752  0.8560856
## carb  0.6773530  0.6127089
##
## $cos2
##          Dim.1      Dim.2
## mpg  0.8685364  0.02688491
## disp 0.8522124  0.06539496
## hp   0.8393393  0.07539375
## wt   0.7559536  0.14263036
## qsec 0.3547632  0.41195966
## gear 0.1500596  0.73288249
## carb 0.4588071  0.37541218
##
## $contrib
##          Dim.1      Dim.2
## mpg  20.294463  1.468673
## disp 19.913033  3.572405
## hp   19.612235  4.118621
## wt   17.663823  7.791632
## qsec  8.289496 22.504591
## gear  3.506334 40.036009
## carb 10.720615 20.508070
```

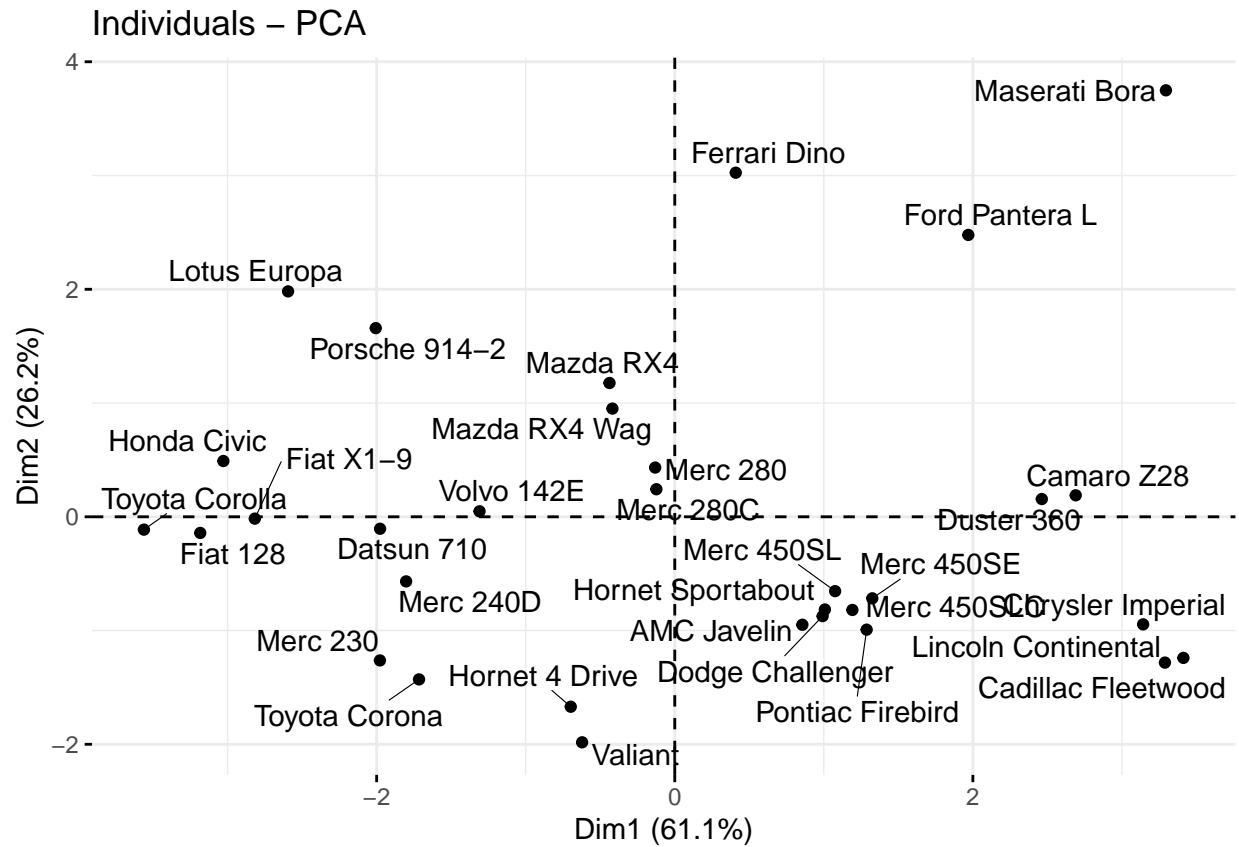
3.3.2 Az eredmények vizualizálása

Az eredmények vizualizálása segíthet a komponensek értelmezésében. A `fviz_pca_var()` és a `fviz_pca_ind()` segítségével reprodukálhatjuk a PCA függő által eredetileg general abrakat. Sőt, a kettőt össze is vonhatjuk a `fviz_pca_biplot()` függővel. Így egyszerre láthatjuk hogy a két legfontosabb dimenzió mentén hol helyezkednek el az egyes megfigyelések (az autók), és hogy a dimenziók főleg mely változókat reprezentálják. (A `repel = T` paraméterbeállítás arra jó hogy a feliratok ne fedjék egymást hanem elcsúsztatva szerepeljenek az ábrán ha túl közel lennének egymáshoz)

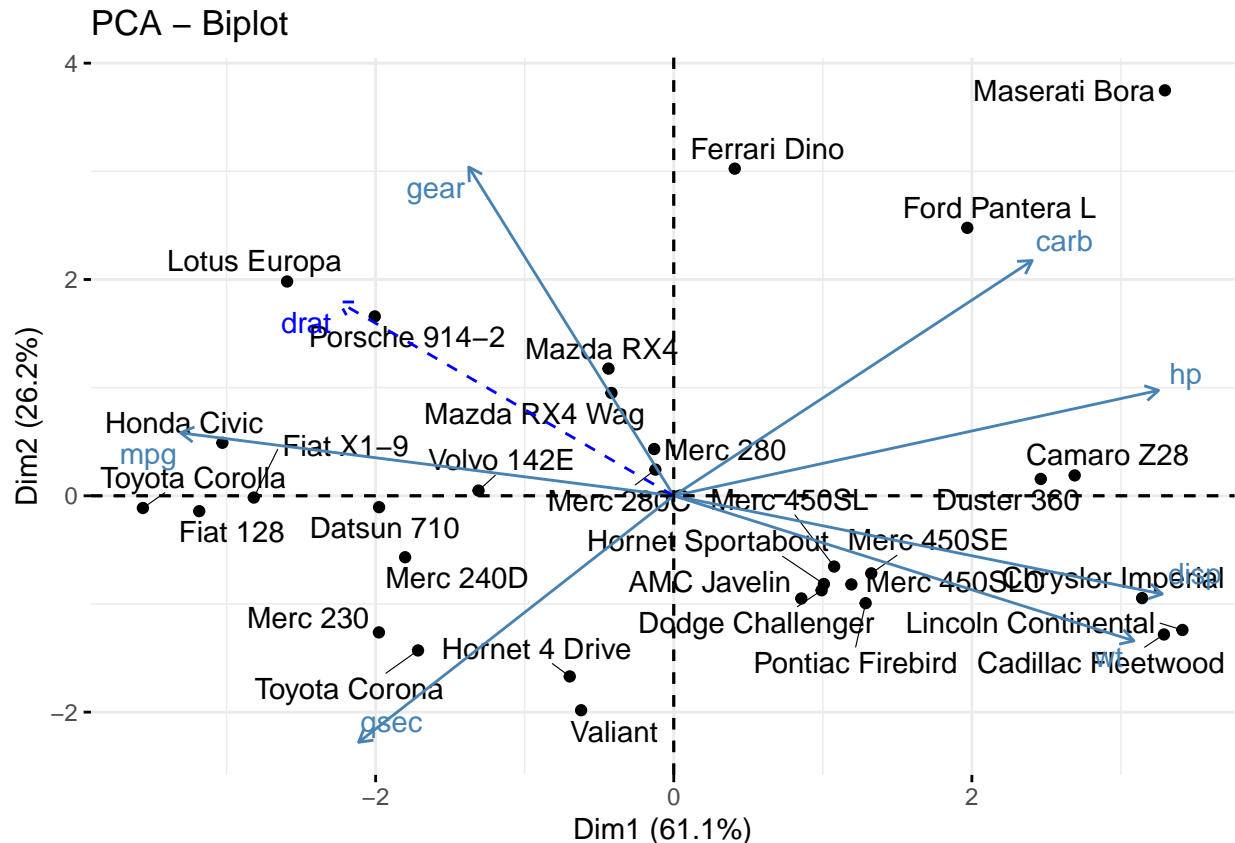
```
fviz_pca_var(pca_mod3, repel = T)
```



```
fviz_pca_ind(pca_mod3, repel = T)
```



```
fviz_pca_biplot(pca_mod3, repel = T)
```

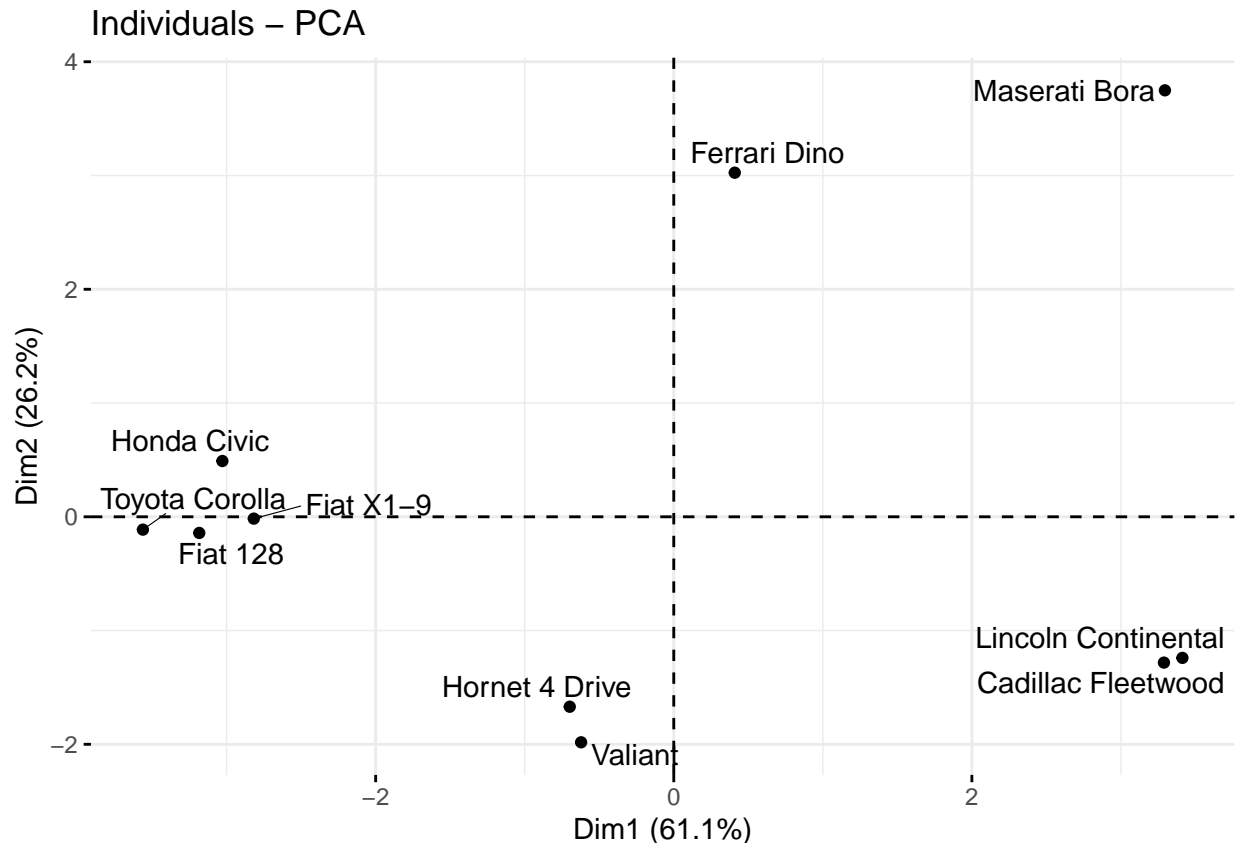


Az abrakat tovább tuningolhatjuk azzal, hogy abrajoljuk rajtuk az egyes változók vagy megfigyelések hozzájárulását (contribution) az abrajolt dimenzióhoz a , col.var = “contrib” és , col.ind = “contrib” paramétereken keresztül.

Azt is megtehetjük, hogy a select.ind = parameteren keresztül hogy csak bizonyos megfigyeléseket teszünk az abrakara.Pl. \cos^2 értékek azt mutatja, hogy az adott megfigyelés vagy változó mennyire jól reprezentált az adott dimenzió által. A select.ind = list(cos2 = 10) parameter beállításával meghatározhatjuk, hogy csak az a 10 megfigyelés szerepeljen az abrakban, akiknek a két dimenzióra vonatkozó \cos^2 összege a legmagasabb. Vagyis a két dimenzió által leírt dimenziótér 10 legreprezentatívabb megfigyelése.

Ez az abrak azt mutatja hogy az alacsony Dim.1, közepes Dim.2 legtipikusabb tagjai pl. a Honda Civic, a Toyota Corolla, a magas Dim.2. közepes Dim.1 legtipikusabb tagja talán a Ferrari Dino, míg a magas Dim.1. és magas Dim.2. legtipikusabb tagja a Maserati Bora. Ez fontos lehet a dimenziók értelmezésében.

```
fviz_pca_ind(pca_mod3, select.ind = list(cos2 = 10), repel = T)
```

Egy másik fontos ábratípus az egyes dimenziók értelmezésének elősegítéséhez a `fviz_contrib()` által generált barchart, ami az egyes változók egyes dimenziókhoz való hozzájárulását mutatja meg. Az `axes =` paraméterrel állíthatjuk be, melyik dimenzióra vagyunk kíváncsiak. A piros szaggatott vonal azt mutatja, hogy mi lenne az elvárt hozzájárulás százaléka abban az esetben ha minden változó azonos mértékben járulna hozzá a dimenzió megmagyarázásához.

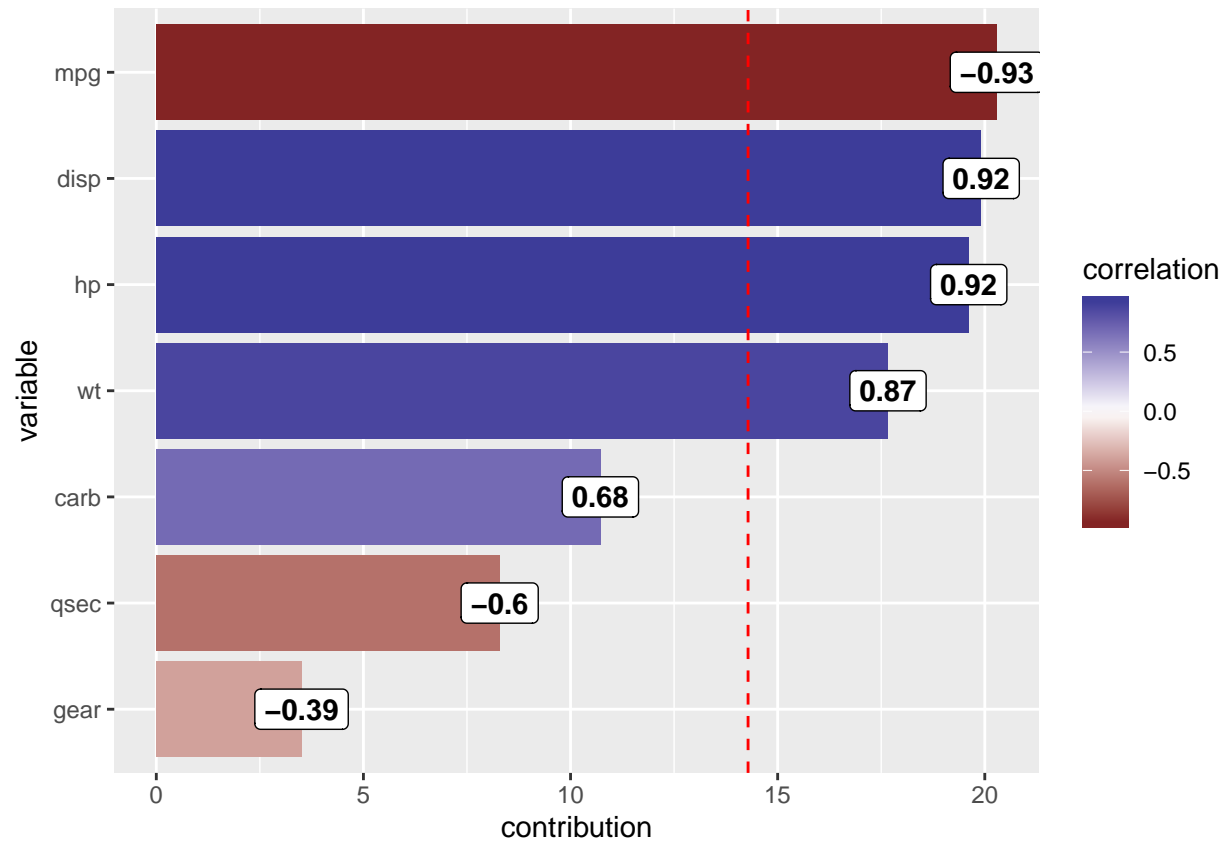
Ez az ábra akkor lenne igazán informatív ha a korreláció mértéke és iránya is egyértelmű lenne. Ezt onmagában nem tartalmazza a `fviz_contrib()` funkció, ezért a `fviz_loadings_with_cor()` saját funkció használatával helyettesítjük, melyen az oszlopok a korreláció szerint vannak színezve és a korreláció felíratként is szerepel az ábrán.

Ezek az ábra azt mutatják, hogy a Dim.1-hez elsősorban az `mpg`, `dist`, `hp`, és `wt` változók járulnak hozzá, míg a Dim.2-hoz elsősorban a `gear` és `qsec` változók járulnak hozzá.

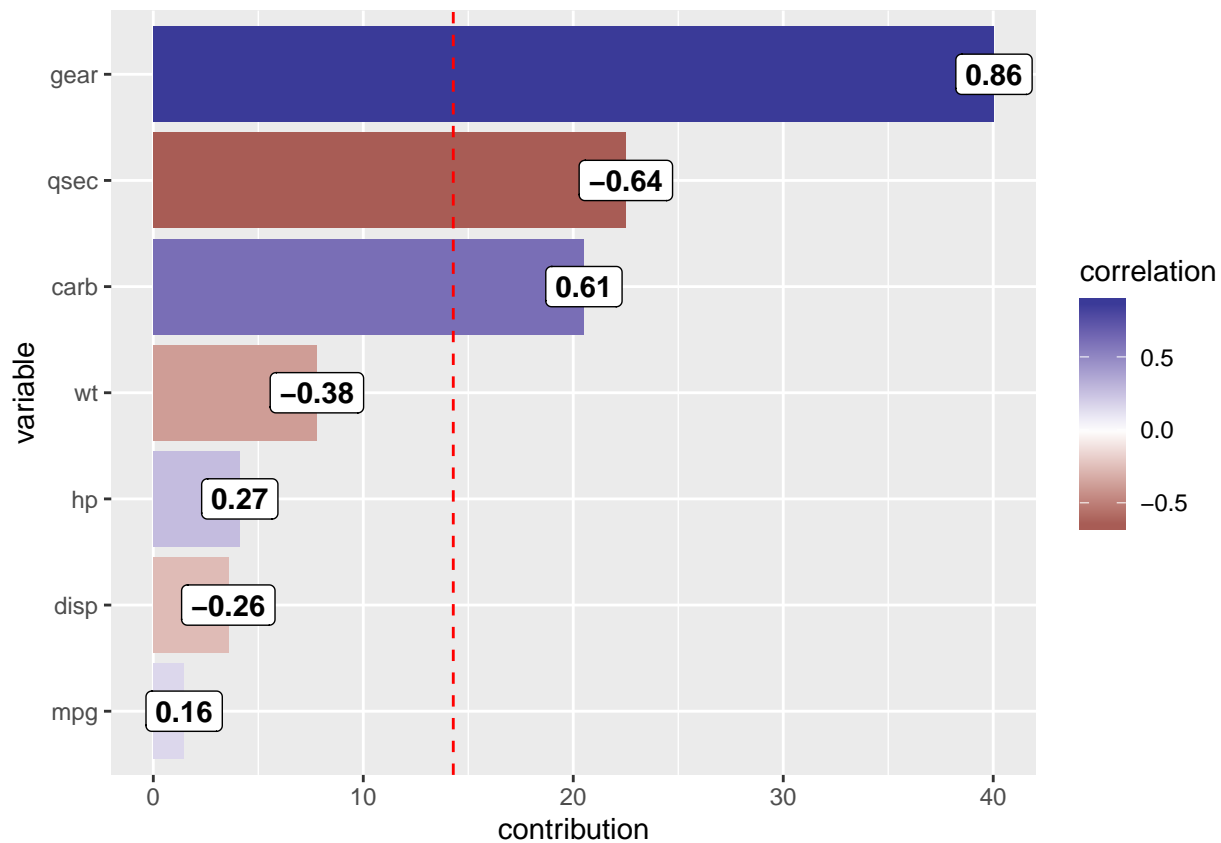
Ezek alapján az ábrák alapján, és a reprezentatív esetek ábrája alapján mit gondolsz, hogyan nevezhetnénk el az egyes és a kettes dimenziót?

```
# original functions in factoextra fviz_contrib(pca_mod3,
# choice = 'var', axes = 1) fviz_contrib(pca_mod3, choice =
# 'var', axes = 2)

# using custom function for correlation color gradient
fviz_loadings_with_cor(mod = pca_mod3, axes = 1)
```



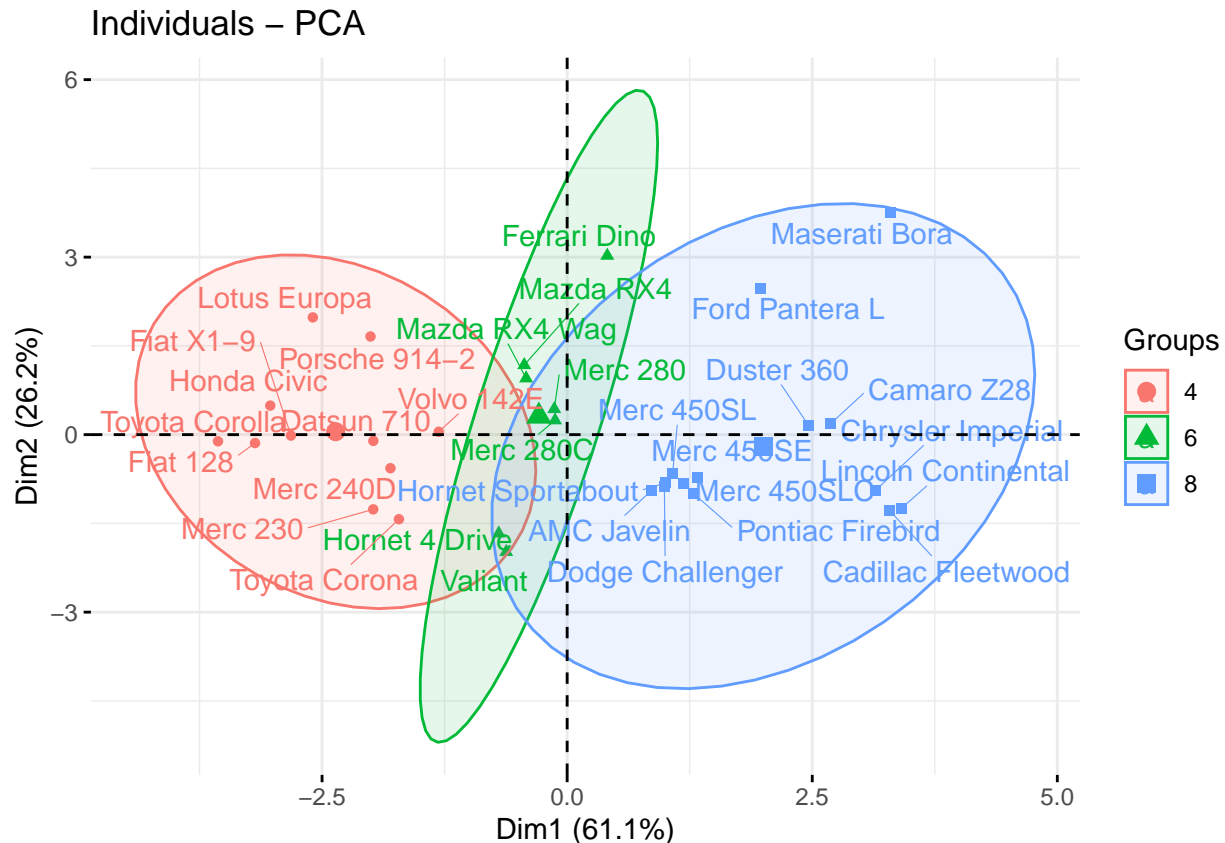
```
fviz_loadings_with_cor(mod = pca_mod3, axes = 2)
```



A vizualizációt arra is használhatjuk, hogy csoportosítsuk a megfigyelesek a dimenziokon mutatott értékek alapján. Ezt az `addEllipses = T` parameterrel adhatjuk meg.

Hogyan jellemeznéd az egyes elipszisekben található autokat az alapján, hogy az 1. és 2. dimenzion milyen értékektől vesznek fel?

```
fviz_pca_ind(pca_mod3, label = "ind", repel = T, habillage = factor(mtcars$cyl),
  addEllipses = T)
```



4 Bevezetés a feltaro faktorelemzesbe (Exploratory Factor Analysis - EFA)

A faktorelemzés egy másik dimenzioredukciós technika, ami hasonlít a főkomponenselemzéshez. A két között fontos különbség, hogy a faktorelemzést akkor használjuk, ha feltételezzük, hogy a sok változónk háttérében közös okok, úgynevezett latens faktorok állnak, és ez okozza, hogy a megfigyelt változókink korrelálnak egymással.

Amikor egy feltáró faktorelemzési (EFA) modellt építünk, nem próbáljuk megmagyarázni a teljes variáciát az adatokban, mert megengedjük, hogy a latens faktorok csak részben magyarázzák a megfigyelt változók variáciáját. A fennmaradó variációt vagy mérési hibát, vagy olyan faktorok befolyását, amelyek egyediak a megfigyelt változóra. Ezért az EFA-ban minden egyes megfigyelt változóra tartozik egy "kommunalitás" (communality) érték. Ez az érték azt mutatja meg, hogy az adott változóban megfigyelhető variáció mekkora hányadát magyarázzák a latens faktorok. A fennmaradó variációt a változó egyedi faktor vagy mérési hiba magyarázza (ezt egyediségnek, vagy uniqueness-nek is nevezzük).

A faktorelemzés legfontosabb lépései:

- Faktoralhatóság ellenőrzése
- Faktorkinyerés
- Ideális faktorszám kiválasztása
- Faktorforgatás
- Faktorok értelmezése

4.1 Új adatok

Alább betöltjük a “Human Styles Questionnaire” adatbázist, ami a Martin et. al. (2003). kutatásából származik, akik a HSQ kerdoivet vettek fel 1071 személlyel.

Az adatbázis első 32 oszlopa Q1-Q32 a kerdoiv egyes teteleire adott válaszokat tartalmazza minden személytől. A válaszadoknak mind a 32 állításról értékelnie kellett, hogy mennyire igaz az rá nézve. A válaszok ordinalis skalan mozognak, 1-től 5-ig: 1=“soha vagy nagyon ritkan igaz”, 5=“nagyon gyakran vagy soha nem igaz”. Ilyen állítások szerepelnek a kerdoivben mint: “Q1: Általában nem nevetek vagy viccelődök másokkal.” (Q1: “I usually don’t laugh or joke around much with other people.”)

A faktorelemzéssel az a célunk, hogy azonosítsuk a háttérben megbúvó pszichológiai vonásokat, amik meghatározzák az egyes embereké hogyan válaszolnak ezekre a tetelekre.

```
hsq <- read_csv("https://raw.githubusercontent.com/kekecsz/PSZB17-210-Data-analysis-seminar/master/seminar_data/hsq.csv")

## Parsed with column specification:
## cols(
##   .default = col_double()
## )

## See spec(...) for full column specifications.
hsq %>% describe()
```

4.2 Adatok faktorálhatósága

Az adatfaktorálhatóság tesztelesekor azt a kérdést válaszoljuk meg, hogy van-e elegendő együttjárás (korreláció) a megfigyelhető változók között, ami lehetővé teszi az EFA elvégzését. Ennek tesztelésére két módszert is alkalmazunk: a Bartlett sphericity tesztet és a Kaiser-Meyer-Olkin tesztet.

Mindenek előtt azonban az adatok korrelációs matrixára van szükségünk, amin ezeket a teszteket lefuttathatjuk. Ezt megkaphatnánk a `cor()` funkcióval ha folytonos változokkal dolgoznánk, de ebben az adatbázisban ordinalis adatokkal van dolgunk, így egy másik funkciót használunk aminek a neve `mixedCor()` a `psych` package-ból. Ez a funkció képes az ordinalis adatok esetén használatos “Polychoric Correlation” meghatározására. A `mixedCor()` funkcióban meghatározzuk hogy melyek a folytonos változók, és melyek az ordinalis változók. A Q1-Q32 mind ordinalis, ezért csak a `p = 1:32`-t határozzuk meg, a `c=t` pedig `NULL`-ra állítjuk, mert nincs folytonos skalan mozgó (continuous) változó.

Fontos, hogy a korrelációs matrixot a `mixedCor()` a `$rho` komponenseben tárolja, ezért ezt kell elmentenünk egy új adatobjektumba. Mentsük el ezt a `hsq_correl` névű objektumba.

```
hsq_mixedCor <- mixedCor(hsq, c = NULL, p = 1:32)

## Warning in matpLower(x, nvar, gminx, gmaxx, gminy, gmaxy): 70 cells were
## adjusted for 0 values using the correction for continuity. Examine your data
## carefully.

hsq_correl = hsq_mixedCor$rho
```

Gyakorlás

A fentebb tanultak alapján vizualizáld a változók közötti korrelációt. Használj több módszert is, pl. `ggcorr()`, `ggcorrplot()` `hc.order=TRUE`-val kombinálva, vagy `network_plot()`.

Bartlett sphericity teszt

A Bartlett teszt lényege hogy a valós korrelációs matrixot összehasonlítjuk egy hipotetikus null-korrelációs matrix-al, amiben minden korreláció 0 értéket vesz fel (identity matrix). A null hipotézis amit itt tesztelünk

az, hogy a két korrelációs matrix nem különbözik egymástól. Ha a teszt szignifikans, az azt jelenti hogy az adattábla változói korrelálnak egymással.

Azonban fontos megjegyezni, hogy a Bartlett tesztnek van egy hátulútje, megpedig hogy nagy elemszámoknál szinte biztosan szignifikans eredményt ad. Csak olyankor érdemes erre a mutatóra hagyatkozni a faktorálhatóság megállapításakor amikor amikor a megfigyelek számok és a megfigyelt változók számának aránya kisebb mint 5. A mi esetünkben ez az arány $1071/32 = 33.5$, vagyis a Bartlett teszt eredménye nem megbízható.

```
bfi_factorability <- cortest.bartlett(hsq_correl)
```

```
## Warning in cortest.bartlett(hsq_correl): n not specified, 100 used
```

```
bfi_factorability
```

```
## $chisq
## [1] 1270.981
##
## $p.value
## [1] 1.861713e-69
##
## $df
## [1] 496
```

Kaiser-Meyer-Olkin (KMO) teszt

A KMO teszt a parciális korrelációs matrixot hasonlítja össze a szokásos korrelációs matrixal. A parciális korreláció során meghatározzuk hogy mekkora két változó közötti korreláció, ha kivonjuk a többi változó hatását a korrelációból. A KMO érték azt mutatja, hogy mekkora a különbség a parciális korrelációk és a szokásos korrelációk között. A KMO egy különbség érték, a parciális korrelációk és a szokásos korrelációk közötti különbséget jelzi. Azokban az esetekben ahol a változók sok közös variációt hordoznak (vagyis valószínű hogy mögöttük egy közös latens faktor áll), a parciális korrelációk alacsonyak, vagyis a KMO index magas. A KMO tesztben az 1-hez közeli értékek jók jó faktorálhatóságot mutatnak. A KMO index-nek legalább 0.6-nak kell lennie hogy úgy ítéljük hogy a változók faktorálhatóak.

A mi példánkban a KMO minden változó esetén magasabb 0.6-nal, és az összesített KMO is magasabb 0.6-nal, így faktorálhatónak tekinthetők a változók.

```
KMO(hsq_correl)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = hsq_correl)
## Overall MSA = 0.88
## MSA for each item =
##   Q1  Q2  Q3  Q4  Q5  Q6  Q7  Q8  Q9  Q10 Q11 Q12 Q13 Q14 Q15 Q16
## 0.94 0.94 0.91 0.90 0.91 0.88 0.83 0.85 0.95 0.86 0.79 0.90 0.87 0.93 0.83 0.88
##   Q17 Q18 Q19 Q20 Q21 Q22 Q23 Q24 Q25 Q26 Q27 Q28 Q29 Q30 Q31 Q32
## 0.90 0.83 0.90 0.82 0.90 0.88 0.83 0.83 0.85 0.90 0.84 0.93 0.88 0.80 0.82 0.91
```

4.3 Faktorextrakció

A faktorokat az `fa()` funkcióval fogjuk kinyerni. Ez a funkció több faktorextrakciós módszert is kínál. A leggyakrabban használt módszer a **Maximum Likelihood Estimation** (mle) akkor ha a megfigyelt változók megfelelnek a többváltozós normalitás feltételének, míg a **Principal Axis Factoring** (paf) a preferált módszer akkor, ha a változók nem mutatnak többváltozós normalis eloszlást.

Az `mvn()` funkció az MVN package-ból és a `mvnrm.kur.test()` és a `mvnrm.skew.test()` funkciók az ICS package-ból segíthet eldönteni, hogy többváltozós normalis eloszlást mutatnak-e az adatok. Ha ezeknek a teszteknek a p-értéke alacsonyabb 0.05-nél, akkor az a többváltozós normalitás sérülésére utal.

```
result <- mvn(hsq[, 1:32], mvnTest = "hz")
result$multivariateNormality
```

```
##           Test           HZ p value MVN
## 1 Henze-Zirkler 1.001426         0 NO
```

```
mvnorm.kur.test(na.omit(hsq[, 1:32]))
```

```
## Warning in pchisqsum(n * W.stat, df = dfs, a = chi.fac, method = "integration"):
## Package 'CompQuadForm' not found, using saddlepoint approximation
```

```
##
## Multivariate Normality Test Based on Kurtosis
##
## data: na.omit(hsq[, 1:32])
## W = 1707.3, w1 = 0.12457, df1 = 527.00000, w2 = 0.23529, df2 = 1.00000,
## p-value < 2.2e-16
```

```
mvnorm.skew.test(na.omit(hsq[, 1:32]))
```

```
##
## Multivariate Normality Test Based on Skewness
##
## data: na.omit(hsq[, 1:32])
## U = 441.69, df = 32, p-value < 2.2e-16
```

Fent látható hogy mind a Henze-Zirkler teszt mind a tobbváltozós ferdeség és csúcsosság tesztek a normalitás feltételének sérülésére utal. Így a paf extrakciós módszert használjuk majd.

A faktorextrakcióra a psych package `fa()` függvényét használjuk. Ezen belül megadhatjuk a faktorextrakciós módszert az `fm` = parameteren belül. Itt `fm` = “pa”-t határozzuk meg, mert a paf módszert szeretnénk használni, de ha a tobbváltozós normalitás nem sérült volna, akkor ehelyett “mle”-t használtunk volna. Az alábbi példában meg nem akartam faktorforgatást alkalmazni, hogy lépésről lépésre tudjam bemutatni a faktorelemzés módszerét, így a `rotate` = értéket “none”-ra állítottam, de általában a faktorokat egyből el is forgatjuk valamelyik módszerrel (lásd alább). Az `nfactors` = parameterrel adhatjuk meg, hány faktort szeretnénk kinyerni. Egyelőre állítsuk ezt 5-re, lentebb tárgyaljuk majd, hogyan választjuk ki az ideális faktormennyiséget.

A modell objektum `$communality` komponensében találjuk a változókhoz tartozó kommunalitás értékeket. Ezt legmagasabbtól legalacsonyabbig sorbarendezzük és kilistázzuk. Ahogy fentebb említettük a kommunalitás azt jelzi, hogy az egyes megfigyelt változókban tapasztalható variancia mekkora hányadát magyarázzak a kinyert faktorok. Az output azt mutatja, hogy a Q17 “Általában nem szeretek viccelődni, vagy másokat szórakoztatni” (“I usually don’t like to tell jokes or amuse people.”) a legjobban reprezentált item az 5 faktoros struktúrában, aminek 68%-át képesek megmagyarázni az új faktorok. Ezzel szemben a Q22 “Amikor szomorú vagy ideges vagyok általában elvesztem a humorérzetemet” (“If I am feeling sad or upset, I usually lose my sense of humor.”) a legkevesbe reprezentált item, varianciájának csak 25%-át magyarázza a jelenlegi faktorstruktúra.

Neha ahhoz hogy a faktorstruktúra jól működjön, érdemes a rosszul reprezentált itemeket kizárni. Ez főleg akkor fontos, ha kicsi a mintaelemszám. Ha a megfigyelések száma 250 alatti, akkor MacCallum et al. szerint elvárható hogy az itemek átlagos kommunalitása legalább 0.6 legyen. A mi esetünkben ennél megengedőbbek is lehetünk, mert az elemszámunk nagyobb, de egy melyebb faktorelemzés esetén így is érdemes lehet elgondolkodni a rosszul reprezentált itemek kizáráson.

MacCallum, R. C., Widaman, K. F., Zhang, S., & Hong, S. (1999). Sample size in factor analysis. *Psychological methods*, 4(1), 84.

```
EFA_mod1 <- fa(hsq_correl, nfactors = 5, fm = "pa")

# Sorted communality
EFA_mod1_common <- as.data.frame(sort(EFA_mod1$communality, decreasing = TRUE))
EFA_mod1_common

##      sort(EFA_mod1$communality, decreasing = TRUE)
## Q17      0.6831095
## Q25      0.6735019
## Q20      0.6686710
## Q21      0.6295625
## Q8       0.6199212
## Q10      0.6152281
## Q18      0.5979830
## Q15      0.5872814
## Q13      0.5586916
## Q14      0.5562796
## Q1       0.5460518
## Q26      0.5392991
## Q32      0.5340582
## Q31      0.5077178
## Q12      0.4800099
## Q19      0.4783614
## Q5       0.4781500
## Q2       0.4558917
## Q16      0.4389957
## Q4       0.4369528
## Q11      0.4056135
## Q6       0.4047964
## Q29      0.3986026
## Q3       0.3968426
## Q7       0.3833806
## Q23      0.3600013
## Q27      0.3476014
## Q24      0.3311687
## Q9       0.3129809
## Q30      0.2661797
## Q28      0.2616243
## Q22      0.2548062

mean(EFA_mod1$communality)

## [1] 0.4752911
```

4.4 Ideális faktorszám kiválasztása

A fokkomponenselemzéshez hasonlóan meg kell határoznunk, hány faktort szeretnénk kinyerni az adatokból. Ahogy azt a fokkomponenselemzésnél is láttuk, ennek az eldöntésére használhatjuk a scree-tesztet, a Kaiser-Guttman kriteriumot, vagy a párhuzamos tesztet. Ezen felül a psych csomag két újabb módszert is felkínál a döntéshozás elősegítésére: a very simple structure (VSS) kriteriumot, és a Wayne Velicer's Minimum Average Partial (MAP) kriteriumot. (A vss() funkció a psych csomagban)

Az alábbi példában a psych csomag fa.parallel függvényét és az nfactors függvényt használjuk arra, hogy a különböző kriteriumok szerint eldönthessük, hány faktor megtartása lenne ideális.

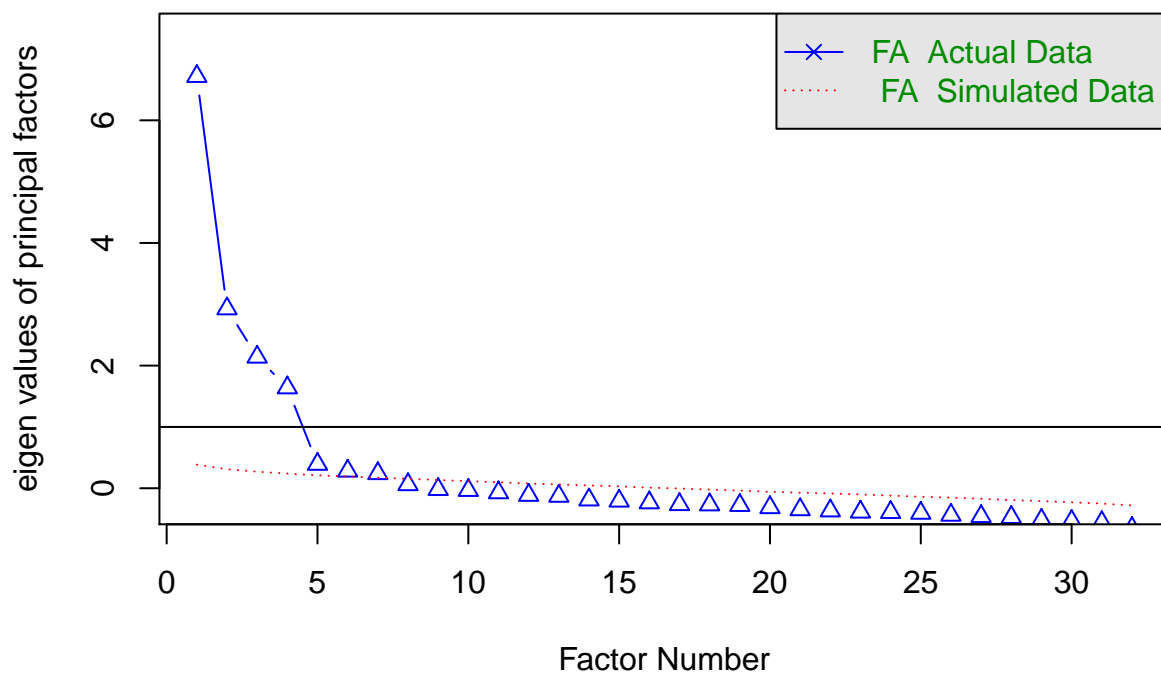
A különböző technikák által javasolt ideális faktorszámok a következők:

- scree-tesztet: 4
- Kaiser-Guttman kriterium: 4
- Parallel tesztet: 7
- VSS: 3-4
- MAP: 4

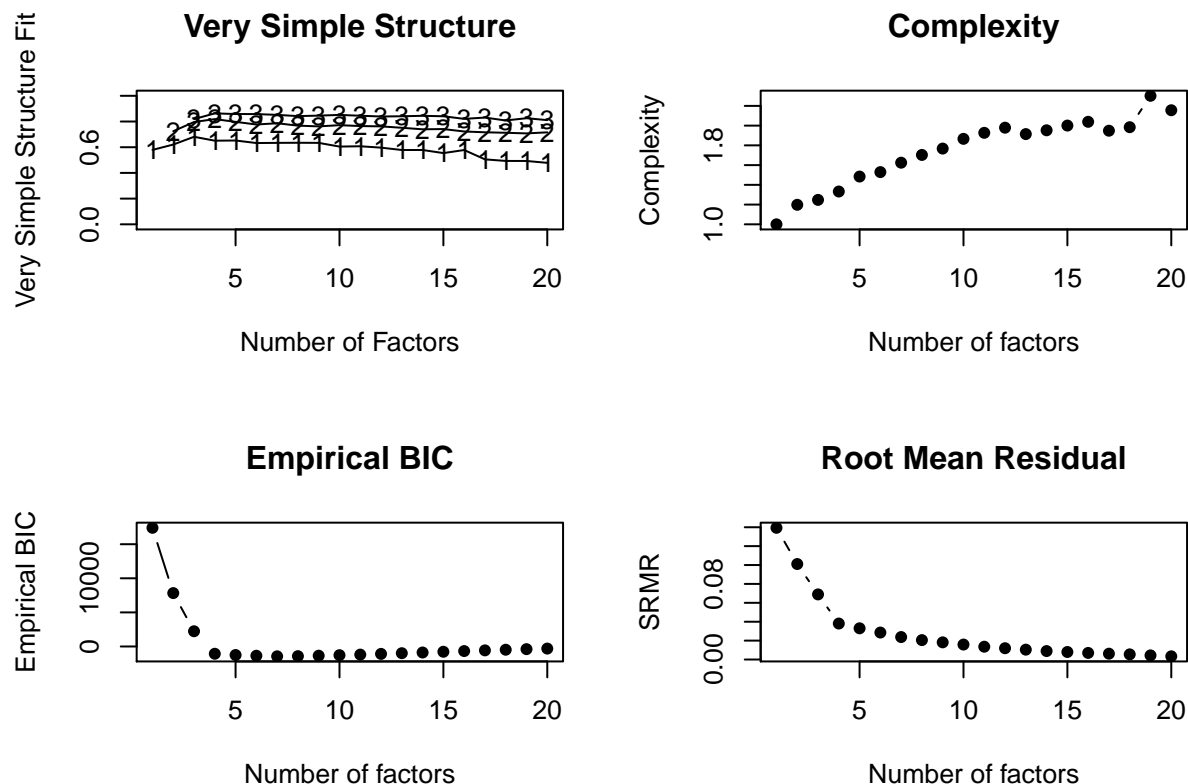
Ezek alapján úgy tűnik, a legtöbb technika szerint 4 latens faktor írja le az adatok variabilitását a legjobban. Alább meg is építjük ezt a 4-faktoros modellt, és megvizsgáljuk a kommunalitás-táblázatot. A faktorelemzés során nagyon gyakori, hogy a folyamatot újra és újra megismételjük különböző bemeneti változokkal és különböző faktorszámokkal és rotációs módszerekkel, amíg elerjük a véglegesnek tekinthető faktorstruktúrát. A végleges faktorstruktúra ideális esetben jól értelmezhető a faktorok és a hozzájuk tartozó változó-töltesek alapján.

```
fa.parallel(hsq_correl, n.obs = nrow(hsq), fa = "fa", fm = "pa")
```

Parallel Analysis Scree Plots



```
## Parallel analysis suggests that the number of factors = 7 and the number of components = NA
nfactors(hsq_correl, n.obs = nrow(hsq))
```



```
##
## 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 maximum of 0.68 with 3 factors
## VSS complexity 2 achieves a maximum of 0.82 with 4 factors
## The Velicer MAP achieves a minimum of 0.01 with 4 factors
## Empirical BIC achieves a minimum of -1443.27 with 7 factors
## Sample Size adjusted BIC achieves a minimum of -211.42 with 14 factors
##
## Statistics by number of factors
##      vss1 vss2  map dof chisq      prob sqresid  fit RMSEA  BIC SABIC complex
## 1  0.58 0.00 0.034 464  9361  0.0e+00   39.8 0.58 0.135 6124  7598    1.0
## 2  0.62 0.72 0.026 433  6451  0.0e+00   26.0 0.72 0.115 3430  4806    1.2
## 3  0.68 0.80 0.019 403  4403  0.0e+00   17.1 0.82 0.097 1592  2872    1.2
## 4  0.65 0.82 0.011 374  2396  2.6e-291  11.6 0.88 0.072 -213   975    1.3
## 5  0.65 0.80 0.012 346  2068  1.7e-242  10.4 0.89 0.069 -346   753    1.5
## 6  0.63 0.78 0.013 319  1712  6.2e-189   9.4 0.90 0.064 -514   500    1.5
## 7  0.63 0.79 0.014 293  1337  8.2e-133   8.5 0.91 0.058 -707   223    1.6
## 8  0.63 0.77 0.016 268  1126  2.1e-105   7.9 0.92 0.055 -744   107    1.7
## 9  0.63 0.76 0.017 244   902  3.1e-76   7.4 0.92 0.051 -800   -25    1.8
## 10 0.60 0.77 0.020 221   734  2.7e-56   6.9 0.93 0.047 -808  -106    1.9
## 11 0.61 0.76 0.022 199   595  4.6e-41   6.5 0.93 0.044 -793  -161    1.9
## 12 0.60 0.76 0.025 178   470  3.3e-28   6.3 0.93 0.040 -772  -206    2.0
## 13 0.58 0.75 0.028 158   397  1.3e-22   5.9 0.94 0.038 -705  -203    1.9
## 14 0.58 0.74 0.032 139   317  6.6e-16   5.6 0.94 0.035 -653  -211    2.0
```

```
## 15 0.55 0.74 0.036 121 281 1.0e-14 5.3 0.94 0.036 -563 -179 2.0
## 16 0.58 0.72 0.040 104 213 1.8e-09 5.2 0.95 0.032 -513 -183 2.0
## 17 0.51 0.72 0.046 88 171 2.8e-07 4.9 0.95 0.030 -443 -163 1.9
## 18 0.49 0.71 0.052 73 133 2.5e-05 4.8 0.95 0.028 -377 -145 2.0
## 19 0.49 0.71 0.059 59 78 5.1e-02 4.4 0.95 0.018 -334 -146 2.3
## 20 0.48 0.72 0.068 46 51 2.7e-01 4.1 0.96 0.012 -270 -123 2.2
## eChisq SRMR eCRMS eBIC
## 1 20643 0.1394 0.144 17406
## 2 10851 0.1011 0.108 7830
## 3 5044 0.0689 0.076 2233
## 4 1539 0.0381 0.044 -1070
## 5 1159 0.0330 0.040 -1254
## 6 866 0.0285 0.036 -1360
## 7 601 0.0238 0.031 -1443
## 8 449 0.0205 0.028 -1421
## 9 351 0.0182 0.026 -1351
## 10 266 0.0158 0.024 -1276
## 11 195 0.0135 0.021 -1193
## 12 152 0.0120 0.020 -1090
## 13 116 0.0104 0.019 -986
## 14 85 0.0089 0.017 -885
## 15 67 0.0080 0.016 -777
## 16 52 0.0070 0.015 -674
## 17 41 0.0062 0.015 -573
## 18 31 0.0054 0.014 -478
## 19 20 0.0044 0.013 -391
## 20 12 0.0034 0.011 -309
```

```
EFA_mod2 <- fa(hsq_correl, nfactors = 4, fm = "pa")
```

```
EFA_mod2_common <- as.data.frame(sort(EFA_mod2$communality, decreasing = TRUE))
EFA_mod2_common
```

```
## sort(EFA_mod2$communality, decreasing = TRUE)
## Q17 0.6849707
## Q20 0.6693961
## Q25 0.6673788
## Q21 0.6258319
## Q8 0.6221044
## Q10 0.6182339
## Q18 0.5993342
## Q14 0.5485758
## Q1 0.5479302
## Q13 0.5457372
## Q26 0.5417358
## Q32 0.5372812
## Q15 0.5189649
## Q31 0.5091859
## Q12 0.4810001
## Q5 0.4530531
## Q2 0.4364573
## Q4 0.4315326
## Q3 0.4003995
## Q29 0.3980778
## Q7 0.3828435
```

```
## Q16 0.3756515
## Q19 0.3686066
## Q6 0.3584248
## Q27 0.3444689
## Q11 0.3293748
## Q9 0.3062011
## Q23 0.2801839
## Q24 0.2796452
## Q30 0.2484149
## Q28 0.2312722
## Q22 0.1850645
```

```
mean(EFA_mod2$communality)
```

```
## [1] 0.4539792
```

4.5 Faktorforgatas

A faktorforgatas celja hogy megkonnyitse a faktorok értelmezését. Így elkerülhető hogy az egész faktorstruktúra 1 vagy két nagyon domináns faktorból álljon, amire angyon erősek a töltések, míg a többi faktor értelmezése kódos. A faktorforgatas során az eredeti változók ugyan ott maradnak a “faktorterben”, viszont a faktorok dimenzio tengelyeit elforgatjuk, hogy jobban railleszkedjenek egyes változócsoportokra.

A faktorforgatasnak számos módszere ismert, de ezek két fő csoportba sorolhatók: ortogonális és oblique módszerek közé. Az ortogonális módszerek (mint pl. Quartimax, Equimax, vagy a pszichológiában leggyakrabban használt **Varimax** módszer) során a faktor dimenziók egymásra merőlegesek maradnak (ez azt jelenti hogy egymással nem korrelálnak majd a végso faktorok). Az oblique módszerek (mint pl. **Direct Oblimin** vagy a Promax) esetén viszont megengedett hogy a végso faktorok valamelyest korreláljanak egymással. Az exploratoros faktorelemzés során több módszert is kipróbálhatunk, de itt fontos az elméleti megalapozottság is. Elképzeltető hogy a faktorok korreláljanak egymással? Ha igen, akkor az oblique módszerekre érdemes hagyatkozni. (Általában a korrelálatlan faktorokat könnyebb értelmezni).

Az alapértelmezett faktorforgatási módszer a Direct Oblimin (“oblimin”). Próbáljuk ki a Promax (“promax”) és a Varimax (“varimax”) módszereket is.

```
EFA_mod2$rotation
```

```
## [1] "oblimin"
```

```
EFA_mod_promax <- fa(hsq_correl, nfactors = 4, fm = "pa", rotate = "promax")
```

```
EFA_mod_varimax <- fa(hsq_correl, nfactors = 4, fm = "pa", rotate = "varimax")
```

4.6 Faktorok interpretacioja

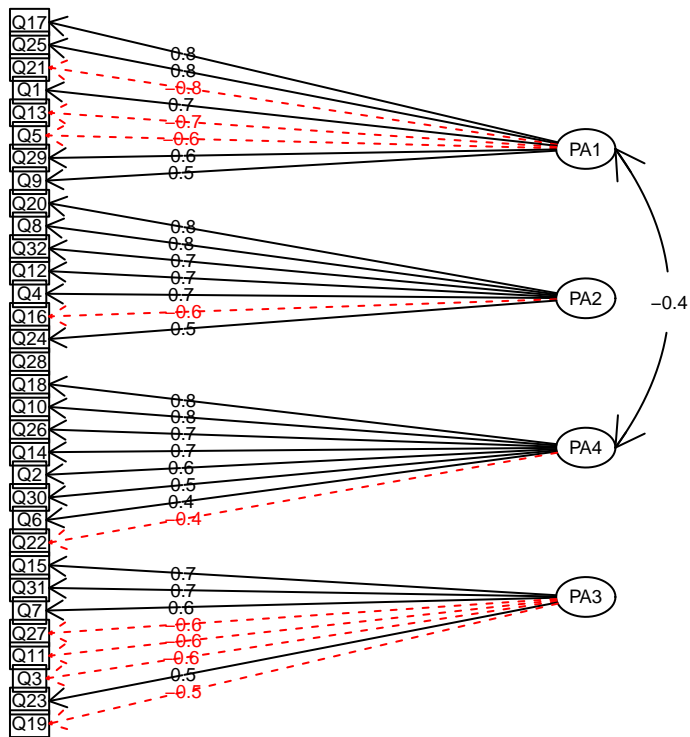
A faktorok értelmezése nem könnyű feladat. Sok területspecifikus tudásra van szükség a helyes faktórsztruktúra kiválasztásához és a helyes faktorértelmezéshez. Itt ezért csak a különböző vizualizációs módszereket mutatjuk be amik segíthetnek a faktorok értelmezésében.

Az `fa.diagram()` funkció kirajzolja a model objektum alapján a faktorstruktúrát, és azt, hogy melyik változó melyik faktorra mutatja a legnagyobb faktortöltést (melyik faktossal a legnagyobb korrelációja). Az ábrán láthatóak az egyes korrelációs együtthatók is. A fekete nyilak pozitív, míg a piros nyilak negatív korrelációkat jeleznek.

További segítséget nyújthat a saját funkció amit a főkomponens-elemzésnél is használtunk: `fviz_loadings_with_cor()`. Itt a `fa()` modellek esetén megadhatjuk a `loading_above =` paramétert is, ahol specifikálhatjuk, hogy csak a bizonyos abszolút faktortöltés (korreláció) feletti megfigyelt változókat ábrázoljuk. Ez megkönnyíti az ábra átlathatóságát.

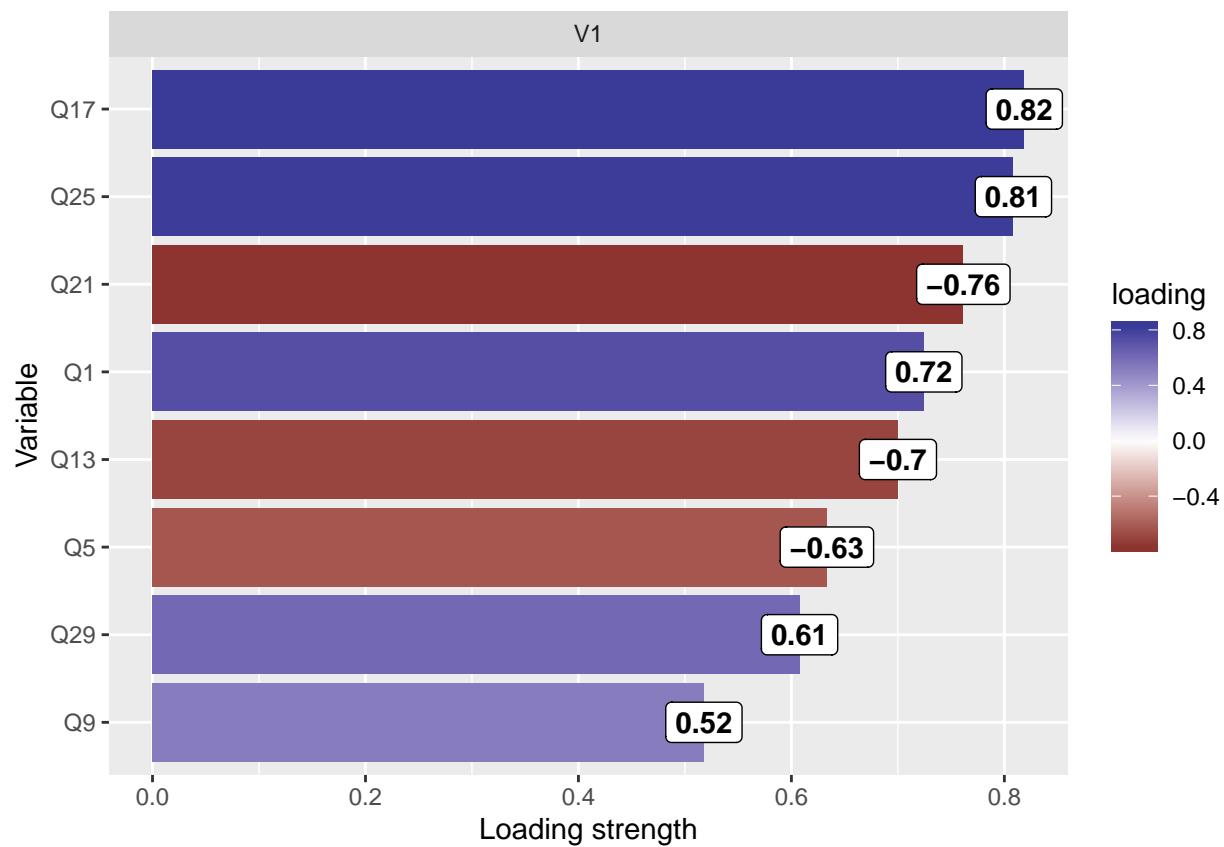
```
fa.diagram(EFA_mod2)
```

Factor Analysis

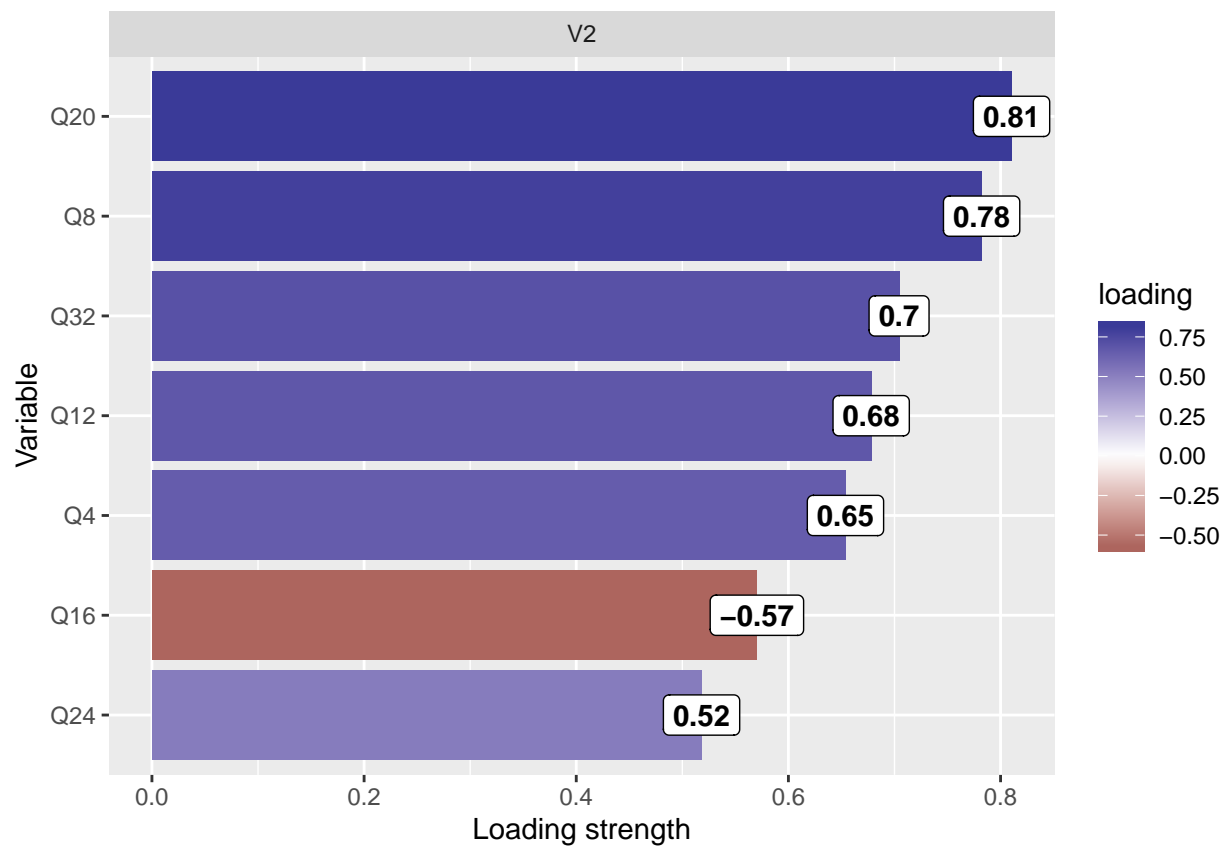


```
fviz_loadings_with_cor(EFA_mod2, axes = 1, loadings_above = 0.4)
```

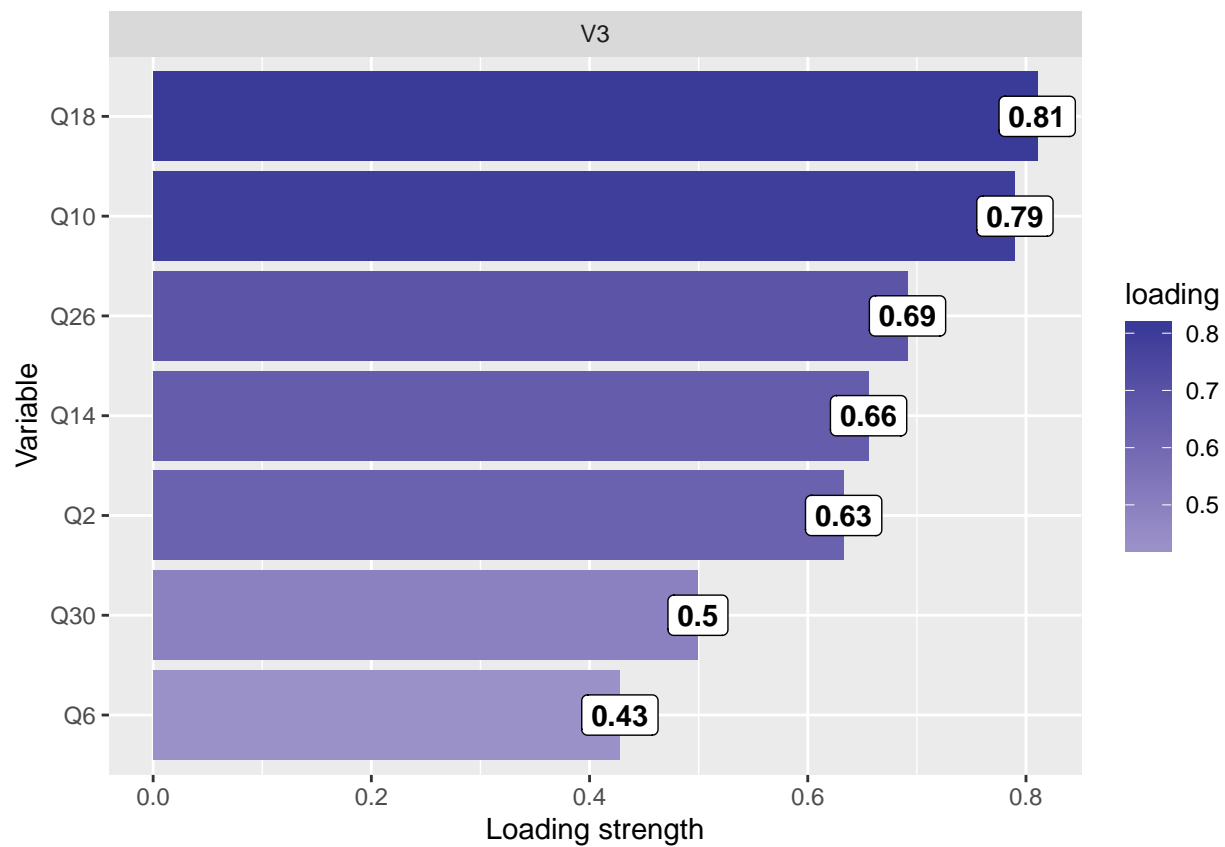
```
## Warning: `as_tibble.matrix()` requires a matrix with column names or a `.name_repair` argument. Using
## This warning is displayed once per session.
```



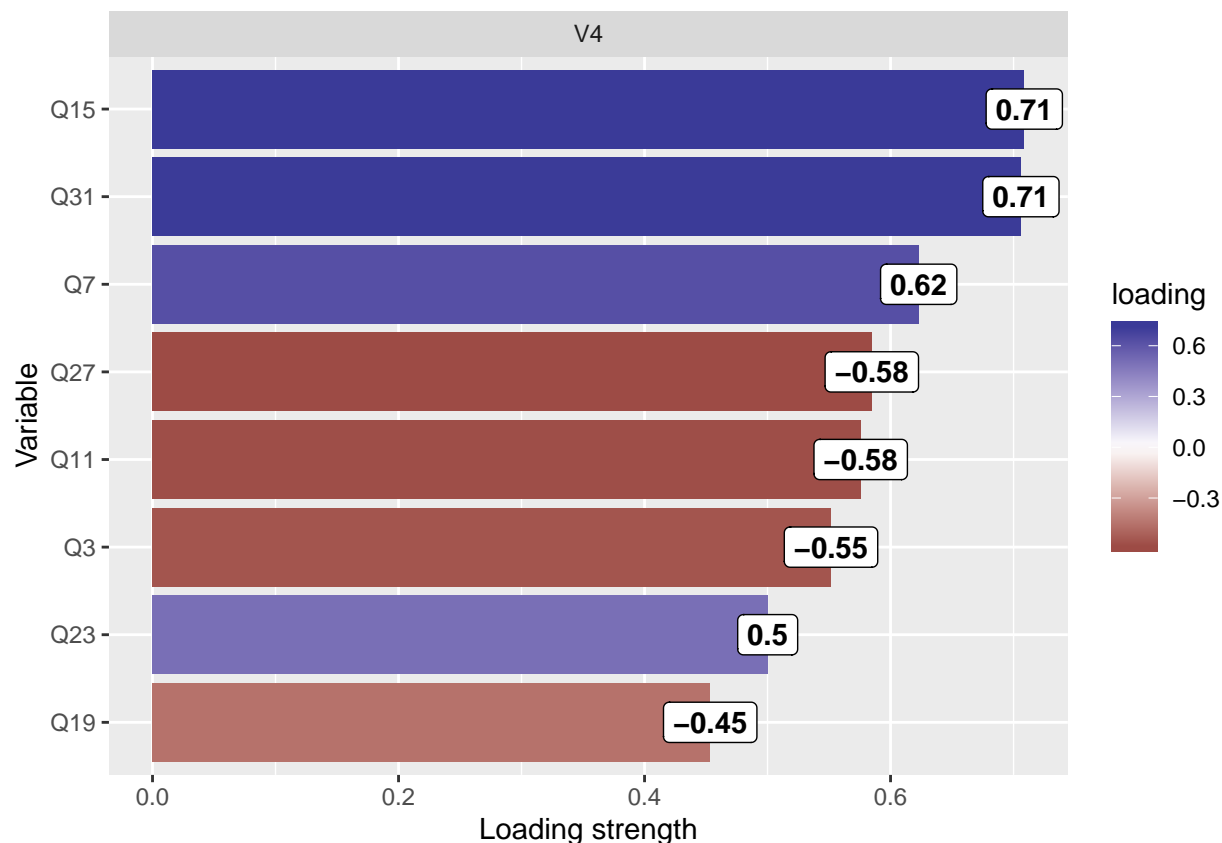
```
fviz_loadnings_with_cor(EFA_mod2, axes = 2, loadings_above = 0.4)
```



```
fviz_loadnings_with_cor(EFA_mod2, axes = 3, loadings_above = 0.4)
```



```
fviz_loadnings_with_cor(EFA_mod2, axes = 4, loadings_above = 0.4)
```

Gyakorlas

A fent tanult technikákat a Big Five Inventory (bfi) adatbázison gyakorolhatod. Ez a psych package-be beépített adatbázis, ami 2800 személy válaszait tartalmazza a Big Five személyiségkerdoiv kérdéseire. Az első 25 oszlop a kerdoiv kérdéseire adott válaszokat tartalmazza, az utolsó három oszlop (gender, education, és age) pedig demográfiai kérdéseket tartalmaz. A részleteket az egyes itemekhez tartozó kérdésekről és a válaszok kódolásáról elolvashatod ha lefuttatod a `?bfi` parancsot.

Az adatbázist betöltheted a következő parancsokkal.

```
?(bfi)

data(bfi)
my_data_bfi = bfi[, 1:25]
```

Ebben a feladatban csak az első 25 oszlopot használd, az eredeti kerdoiv kérdéseit. Végezz el feltároló faktorelemzést, és ez alapján határozd meg, hány faktor megtartása az ideális, mely faktorokra mely itemek töltenek leginkább, és ez alapján hogyan nevezned el a faktorokat. Melyek a faktorstruktúra által leginkább és a legkevesbé reprezentált itemek?