Solution for the assignment of the eight class

Kovacs Marton

9/29/2021

Importing data

```r
processed <- read_tsv("data/boldog_processed.tsv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   neme = col_character(),
##   isk = col_character()
## )
## i Use `spec()` for the full column specifications.
```

Data exploration

```r
skimr::skim(processed) %>%
  kable()
```

skim_type	skim_variable	n_missing	complete_rate	character.min	character.max	character.empty	character.n_unique	character.whitespace	numeric.mean	numeric.sd	numeric.p0	numeric.p25	numeric.p50	numeric.p75	numeric.p100	numeric.hist
character	neme	0	1.000	2	5	0	2	0	NA	NA	NA	NA	NA	NA	NA	NA
character	isk	0	1.000	7	11	0	4	0	NA	NA	NA	NA	NA	NA	NA	NA
numeric	index	0	1.000	NA	NA	NA	NA	NA	972.344	[illegible]	[illegible]	598.00	1070.0	1361.75	1700	<U+2585><U+2585><U
numeric	eletkora	0	1.000	NA	NA	NA	NA	NA	52.522	[illegible]	[illegible]	45.000	52.500	61.000	[illegible]	<U+2581><U+2585><U
numeric	gyermeke	0	1.000	NA	NA	NA	NA	NA	1.750	[illegible]	[illegible]	1.000	2.000	2.000	9	<U+2585><U+2587><U
numeric	anyagi	0	1.000	NA	NA	NA	NA	NA	3.100	[illegible]	[illegible]	3.000	3.000	3.000	5	<U+2581><U+2581><U
numeric	pic_elmeny_percent	0	1.000	NA	NA	NA	NA	NA	58.460	[illegible]	[illegible]	40.000	60.000	80.000	[illegible]	<U+2582><U+2585><U
numeric	testi_fi	3	0.994	NA	NA	NA	NA	NA	4.1649	[illegible]	[illegible]	4.000	4.000	5.000	6	<U+2581><U+2583><U
numeric	alt_lelki	4	0.992	NA	NA	NA	NA	NA	4.1612	[illegible]	[illegible]	4.000	4.000	5.000	6	<U+2582><U+2583><U
numeric	alt_eg_all	7	0.986	NA	NA	NA	NA	NA	4.1217	[illegible]	[illegible]	3.000	4.000	5.000	6	<U+2582><U+2583><U
numeric	fizero	6	0.988	NA	NA	NA	NA	NA	4.1538	[illegible]	[illegible]	3.000	4.000	5.000	6	<U+2582><U+2585><U
numeric	aicocska	0	1.000	NA	NA	NA	NA	NA	2.570	[illegible]	[illegible]	2.000	2.000	3.000	7	<U+2587><U+2585><U
numeric	aggodalo	0	1.000	NA	NA	NA	NA	NA	2.772	[illegible]	[illegible]	2.000	2.500	4.000	6	<U+2587><U+2583><U
numeric	ideges	0	1.000	NA	NA	NA	NA	NA	2.490	[illegible]	[illegible]	1.000	2.000	3.000	6	<U+2587><U+2582><U
numeric	feszult	0	1.000	NA	NA	NA	NA	NA	2.610	[illegible]	[illegible]	1.000	2.000	3.000	6	<U+2587><U+2583><U
numeric	nyugtala	0	1.000	NA	NA	NA	NA	NA	2.406	[illegible]	[illegible]	1.000	2.000	3.000	6	<U+2587><U+2582><U

skim_type	skim_variable	n_missing	complete_rate	character.min	character.max	character.empty	character.n_unique	character.whitespace	numeric.mean	numeric.sd	numeric.p0	numeric.p25	numeric.p50	numeric.p75	numeric.p100	numeric.hist
numeric	diener1	0	1.000	NA	NA	NA	NA	NA	5.5140	[illegible]	[illegible]	5.0000	6.0000	6.000000	7	<U+2581><U+2581><U
numeric	diener2	0	1.000	NA	NA	NA	NA	NA	5.2280	[illegible]	[illegible]	4.0000	6.0000	6.000000	7	<U+2581><U+2581><U
numeric	diener3	0	1.000	NA	NA	NA	NA	NA	5.5740	[illegible]	[illegible]	5.0000	6.0000	6.000000	7	<U+2581><U+2581><U
numeric	diener4	0	1.000	NA	NA	NA	NA	NA	5.4180	[illegible]	[illegible]	5.0000	6.0000	6.000000	7	<U+2581><U+2581><U
numeric	diener5	0	1.000	NA	NA	NA	NA	NA	5.7560	[illegible]	[illegible]	5.0000	6.0000	7.000000	7	<U+2581><U+2581><U
numeric	diener6	0	1.000	NA	NA	NA	NA	NA	5.7680	[illegible]	[illegible]	5.0000	6.0000	7.000000	7	<U+2581><U+2581><U
numeric	diener7	0	1.000	NA	NA	NA	NA	NA	5.6200	[illegible]	[illegible]	5.0000	6.0000	7.000000	7	<U+2581><U+2581><U
numeric	diener8	0	1.000	NA	NA	NA	NA	NA	5.5320	[illegible]	[illegible]	5.0000	6.0000	6.000000	7	<U+2581><U+2581><U
numeric	jollet	0	1.000	NA	NA	NA	NA	NA	4.4886	[illegible]	[illegible]	4.0000	4.6667	5.333300	6	<U+2581><U+2582><U
numeric	savor	0	1.000	NA	NA	NA	NA	NA	4.5146	[illegible]	[illegible]	4.0000	4.6667	5.000000	6	<U+2581><U+2582><U
numeric	a_vhat	0	1.000	NA	NA	NA	NA	NA	4.5956	[illegible]	[illegible]	4.0000	4.8000	5.200000	6	<U+2581><U+2581><U
numeric	oireg	0	1.000	NA	NA	NA	NA	NA	4.2173	[illegible]	[illegible]	3.3333	4.3333	5.083325	6	<U+2582><U+2583><U
numeric	rezil	0	1.000	NA	NA	NA	NA	NA	4.0446	[illegible]	[illegible]	3.3333	4.0000	4.666700	6	<U+2581><U+2583><U
numeric	m_flow	0	1.000	NA	NA	NA	NA	NA	4.7739	[illegible]	[illegible]	4.3333	5.0000	5.333300	6	<U+2581><U+2581><U
numeric	g_jerz	0	1.000	NA	NA	NA	NA	NA	4.1528	[illegible]	[illegible]	3.4000	4.2000	5.000000	6	<U+2581><U+2583><U
numeric	g_jpszi	0	1.000	NA	NA	NA	NA	NA	4.2820	[illegible]	[illegible]	3.7500	4.5000	5.000000	6	<U+2581><U+2582><U
numeric	g_jszoc	0	1.000	NA	NA	NA	NA	NA	3.9495	[illegible]	[illegible]	3.0000	4.0000	5.000000	6	<U+2582><U+2585><U
numeric	g_jspir	0	1.000	NA	NA	NA	NA	NA	4.2455	[illegible]	[illegible]	3.5000	4.2500	5.250000	6	<U+2582><U+2583><U
numeric	g_jerzpsz	0	1.000	NA	NA	NA	NA	NA	4.2174	[illegible]	[illegible]	3.6187	4.3375	4.925000	6	<U+2581><U+2582><U
numeric	g_jspszoc	0	1.000	NA	NA	NA	NA	NA	4.0975	[illegible]	[illegible]	3.3750	4.1250	5.000000	6	<U+2581><U+2583><U
numeric	pik_mm	0	1.000	NA	NA	NA	NA	NA	3.0273	[illegible]	[illegible]	2.6667	3.1667	3.500000	4	<U+2581><U+2582><U
numeric	pik_av	0	1.000	NA	NA	NA	NA	NA	3.1750	[illegible]	[illegible]	2.7500	3.2500	3.750000	4	<U+2581><U+2581><U
numeric	pik_onr	0	1.000	NA	NA	NA	NA	NA	3.0640	[illegible]	[illegible]	2.6667	3.3333	3.666700	4	<U+2581><U+2582><U
numeric	pik_rez	0	1.000	NA	NA	NA	NA	NA	3.1506	[illegible]	[illegible]	2.6667	3.3333	3.666700	4	<U+2581><U+2582><U
numeric	p_poz_erz	0	1.000	NA	NA	NA	NA	NA	21.742	[illegible]	[illegible]	20.000	23.000	26.000	30	<U+2581><U+2581><U
numeric	p_elmely	0	1.000	NA	NA	NA	NA	NA	22.360	[illegible]	[illegible]	20.000	23.000	26.000	30	<U+2581><U+2582><U
numeric	p_poz_kapc	0	1.000	NA	NA	NA	NA	NA	22.442	[illegible]	[illegible]	19.000	24.000	26.000	30	<U+2581><U+2581><U
numeric	p_ert_cel	0	1.000	NA	NA	NA	NA	NA	23.374	[illegible]	[illegible]	21.000	24.000	27.000	30	<U+2581><U+2581><U
numeric	p_telj	0	1.000	NA	NA	NA	NA	NA	22.716	[illegible]	[illegible]	21.000	24.000	26.000	30	<U+2581><U+2581><U
numeric	p_boldog	0	1.000	NA	NA	NA	NA	NA	7.4040	[illegible]	[illegible]	7.0000	8.0000	9.000000	10	<U+2581><U+2581><U
numeric	p_egeszs	0	1.000	NA	NA	NA	NA	NA	22.260	[illegible]	[illegible]	19.000	23.000	26.000	30	<U+2581><U+2581><U
numeric	p_neg_erz	0	1.000	NA	NA	NA	NA	NA	9.7660	[illegible]	[illegible]	5.0000	9.0000	13.000	30	<U+2587><U+2587><U
numeric	p_magany	0	1.000	NA	NA	NA	NA	NA	3.1620	[illegible]	[illegible]	0.0000	2.0000	5.000000	10	<U+2587><U+2582><U
numeric	perma	0	1.000	NA	NA	NA	NA	NA	169.37	[illegible]	[illegible]	155.00	[illegible]	191.00	[illegible]	<U+2581><U+2581><U

1. PCA on Diener flourishing scale

First, we select the variables of interest.

```
diener_data <- dplyr::select(processed, contains("diener"))
```

In R, we can run the PCA with just one function. We can set the scale argument to true to scale our variables to unit variance and center the variables.

```
pca <-
  diener_data %>%
  prcomp(scale = TRUE, center = TRUE)

summary(pca)
```

```
## Importance of components:
##                           PC1     PC2     PC3    PC4     PC5     PC6     PC7
## Standard deviation     2.3837 0.70037 0.67176 0.6203 0.55088 0.54801 0.46048
## Proportion of Variance 0.7102 0.06131 0.05641 0.0481 0.03793 0.03754 0.02651
## Cumulative Proportion  0.7102 0.77154 0.82795 0.8760 0.91399 0.95153 0.97803
##                           PC8
## Standard deviation     0.41922
## Proportion of Variance 0.02197
## Cumulative Proportion  1.00000
```

As the data are standardized the standard deviation values for each principal component are the Eigenvalues.

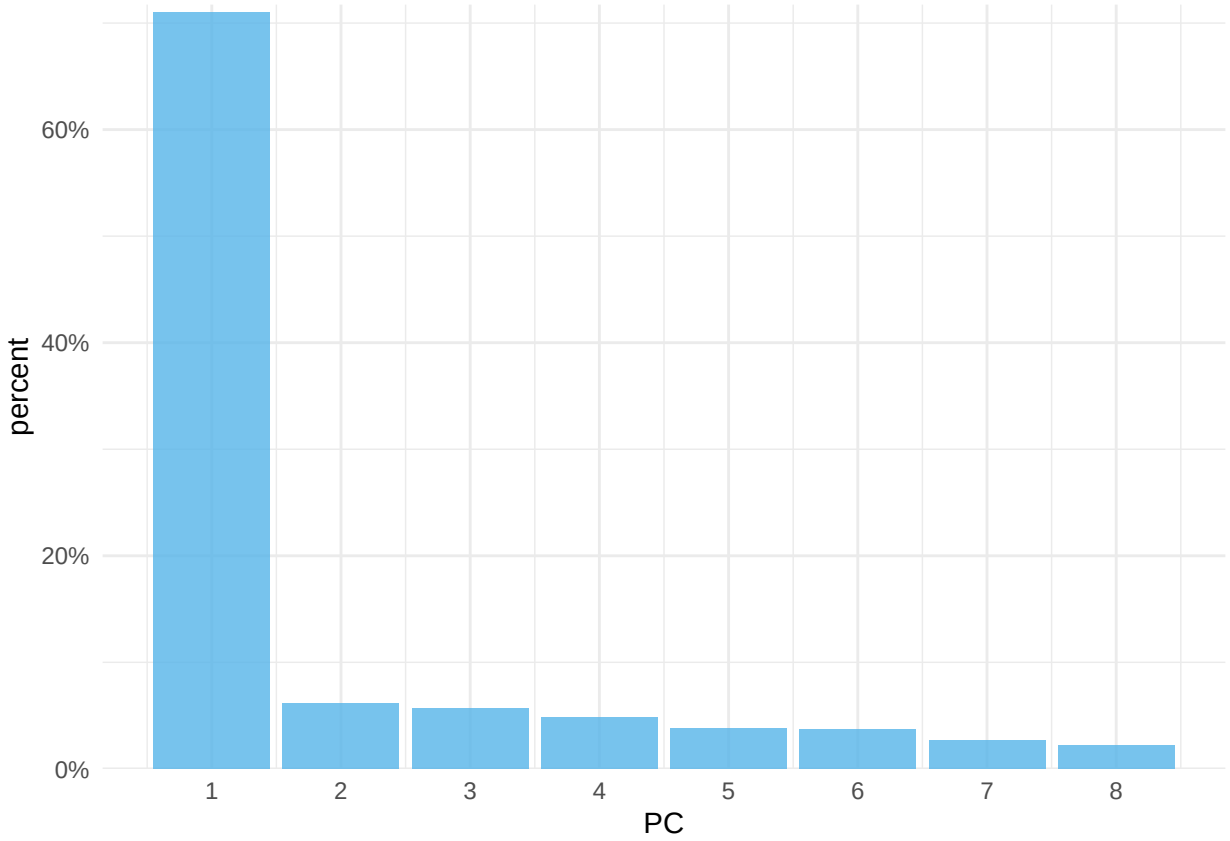
As we can see only the first PC has an Eigenvalue greater than 1.

The first PC accounts for 71.02% of the variance, while the second PC accounts for the 6.13% of the variance.

Together they explain 77.15% of the variance.

Lets plot the results.

```
pca %>%
  tidy(matrix = "eigenvalues") %>%
  ggplot(aes(PC, percent)) +
  geom_col(fill = "#56B4E9", alpha = 0.8) +
  scale_x_continuous(breaks = 1:8) +
  scale_y_continuous(
    labels = scales::percent_format(),
    expand = expansion(mult = c(0, 0.01))
  ) +
  theme_minimal()
```



2. PCA of MET

First, we select the variables of interest.

```
met_subscales <- c("jollet", "savor", "a_vhat", "onreg", "rezil")

met_data <- dplyr::select(processed, all_of(met_subscales))
```

We run the PCA while standardizing the variables.

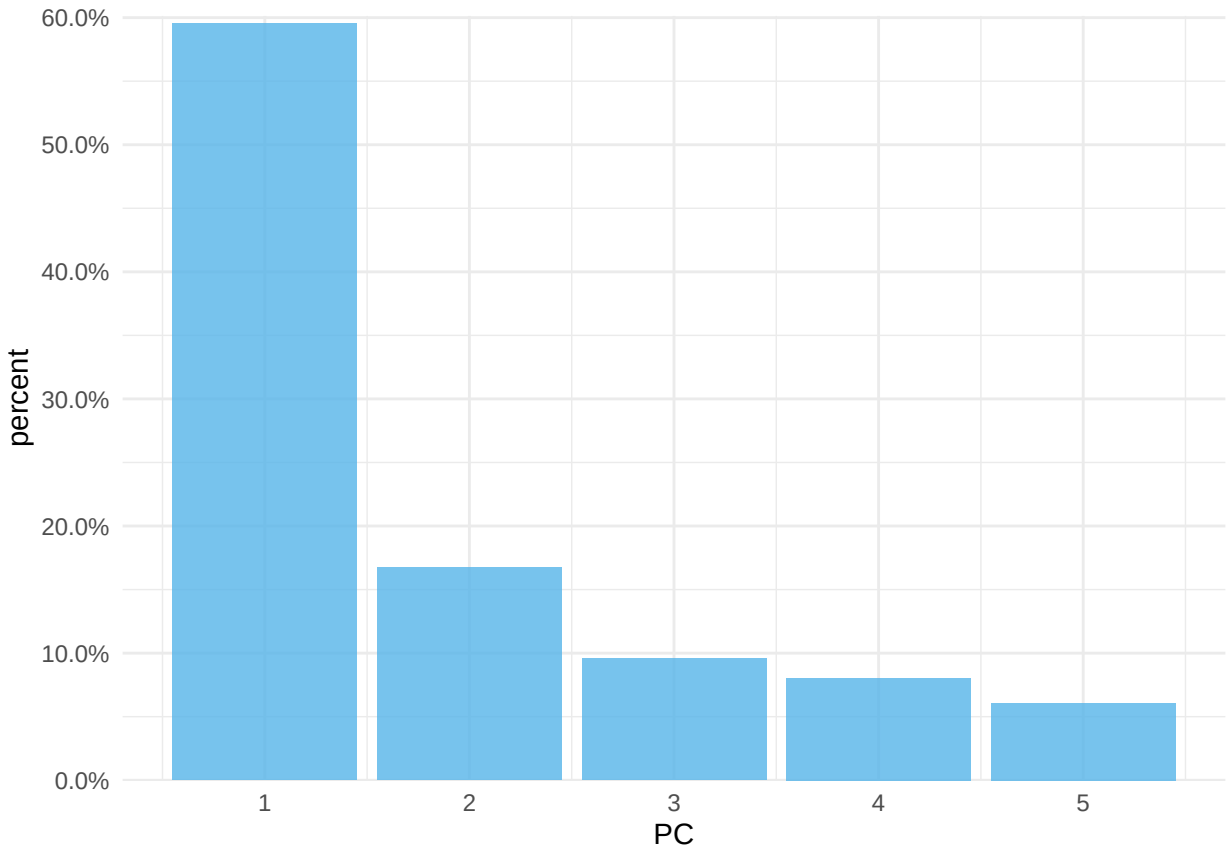
```
pca_met <-
  met_data %>%
  prcomp(scale = TRUE, center = TRUE)

summary(pca_met)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5
## Standard deviation     1.7254 0.9153 0.69228 0.63458 0.55085
## Proportion of Variance 0.5954 0.1676 0.09585 0.08054 0.06069
## Cumulative Proportion  0.5954 0.7629 0.85878 0.93931 1.00000
```

As we can see only the first PC has an Eigenvalue greater than 1, and it accounts for 59.54% of the variance.

```r
pca_met %>%
  tidy(matrix = "eigenvalues") %>%
  ggplot(aes(PC, percent)) +
  geom_col(fill = "#56B4E9", alpha = 0.8) +
  scale_x_continuous(breaks = 1:5) +
  scale_y_continuous(
    labels = scales::percent_format(),
    expand = expansion(mult = c(0, 0.01))
  ) +
  theme_minimal()
```



3. Use varimax and promax rotation on the first three principal components of the second task.

I could use the `prcomp` function ad before, but we would have to use a cutting point for the Eigenvalues to only include 3 PC. Thus, I will use the `psych::principal` function, where I can set the number of PC in the function call.

Thus, I run a new PCA with three components and varimax rotation. This function returns the standardized scores by default. The function calculates the correlation matrix by pairwise deletion of the missing values.

```r
pca_met_varimax <- psych::principal(met_data, rotate = "varimax", nfactors = 3, scores = TRUE)
```

Lets take a look at the Eigenvalues.

```
pca_met_varimax$value
```

```
## [1] 2.9768986 0.8377353 0.4792455 0.4026888 0.3034318
```

The rest of the results.

```
pca_met_varimax
```

```
## Principal Components Analysis
## Call: psych::principal(r = met_data, nfactors = 3, rotate = "varimax",
##     scores = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##         RC1  RC3  RC2   h2    u2 com
## jollet 0.67 0.55 0.14 0.77 0.230 2.0
## savor  0.91 0.10 0.12 0.86 0.144 1.1
## a_vhat 0.74 0.45 0.12 0.76 0.240 1.7
## onreg  0.14 0.22 0.97 1.00 0.002 1.1
## rezil  0.24 0.89 0.25 0.91 0.090 1.3
##
##                        RC1  RC3  RC2
## SS loadings           1.90 1.36 1.04
## Proportion Var        0.38 0.27 0.21
## Cumulative Var        0.38 0.65 0.86
## Proportion Explained  0.44 0.32 0.24
## Cumulative Proportion 0.44 0.76 1.00
##
## Mean item complexity =  1.5
## Test of the hypothesis that 3 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.08
##  with the empirical chi square  57.46  with prob <  NA
##
## Fit based upon off diagonal values = 0.98
```

The factor loadings are quite small for *onreg* and *rezil* on RC1. Based on the communalities (h2) it seems that most of the variance explained by these two items can be explained by other items.

The complexity scores suggest that *jollet* and maybe *a_vhat* the items measure more than one latent construct.

The proportion var suggest that the RC1 explains 38% of the variance in the responses, while RC3 27% and RC2 21%.

The fit based upon off diagonal values support a good fit.

Now we should run the same analysis but allow factors to correlate, therefore we use an oblique rotation called promax.

```
pca_met_promax <- psych::principal(met_data, rotate = "promax", nfactors = 3, scores = TRUE)
```

Lets see the results of this solution.

```
pca_met_promax
```

```
## Principal Components Analysis
## Call: psych::principal(r = met_data, nfactors = 3, rotate = "promax",
##     scores = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##          RC1   RC3   RC2   h2    u2 com
## jollet  0.54  0.43 -0.03 0.77 0.230 1.9
## savor   1.07 -0.27  0.04 0.86 0.144 1.1
## a_vhat  0.68  0.27 -0.03 0.76 0.240 1.3
## onreg   0.02  0.04  0.97 1.00 0.002 1.0
## rezil  -0.16  1.04  0.04 0.91 0.090 1.0
##
##                        RC1  RC3  RC2
## SS loadings           1.92 1.40 0.98
## Proportion Var        0.38 0.28 0.20
## Cumulative Var        0.38 0.66 0.86
## Proportion Explained  0.45 0.33 0.23
## Cumulative Proportion 0.45 0.77 1.00
##
##  With component correlations of
##      RC1  RC3  RC2
## RC1 1.00 0.66 0.29
## RC3 0.66 1.00 0.42
## RC2 0.29 0.42 1.00
##
## Mean item complexity =  1.3
## Test of the hypothesis that 3 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.08
##  with the empirical chi square  57.46  with prob <  NA
##
## Fit based upon off diagonal values = 0.98
```

Allowing for correlation reduced the complexity and we have a much cleaner latent structure than before with varimax. RC1 highly correlates (0.66) with RC3 and only medium wiht RC2 (0.29), there is a medium to high correlation (0.42) between RC3 and RC2.

4. Save the factors from the promax rotation and correlate the with the PERMA test's items. Which items has the highest correlattion with the first factor.

Calculating the factor scores.

```
scores <- factor.scores(met_data, pca_met_promax)

scores <- as.data.frame(scores$scores)
```

Binding calculated scores to the original data.

```
perma_vars <- c("p_poz_erz", "p_elmely", "p_poz_kapc", "p_ert_cel", "p_telj", "p_boldog", "p_egeszs", "p

processed_scores <-
  processed %>%
  bind_cols(., scores) %>%
  select(RC1, RC2, RC3, all_of(perma_vars))
```

Running the correlations.

```
corr_matrix <- rcorr(as.matrix(processed_scores))

corr_matrix$r
```

```
##                  RC1        RC2        RC3  p_poz_erz   p_elmely p_poz_kapc
## RC1        1.0000000  0.2864299  0.6621899  0.6913069  0.4993597  0.5658908
## RC2        0.2864299  1.0000000  0.4246157  0.4356578  0.2492797  0.2877081
## RC3        0.6621899  0.4246157  1.0000000  0.7040195  0.4838113  0.5060462
## p_poz_erz  0.6913069  0.4356578  0.7040195  1.0000000  0.5775692  0.6774156
## p_elmely   0.4993597  0.2492797  0.4838113  0.5775692  1.0000000  0.4796039
## p_poz_kapc 0.5658908  0.2877081  0.5060462  0.6774156  0.4796039  1.0000000
## p_ert_cel  0.6604940  0.2450133  0.6050290  0.7531407  0.6305894  0.6495659
## p_telj     0.6053890  0.2457362  0.5759880  0.7229889  0.5485736  0.5369546
## p_boldog   0.6577439  0.3347994  0.6490360  0.8331380  0.5095568  0.7279601
## p_egeszs   0.4361927  0.2566204  0.4511676  0.5274025  0.3379240  0.4356007
## p_neg_erz -0.5526917 -0.4805750 -0.6201278 -0.6866772 -0.4282924 -0.4972611
## p_magany  -0.3836024 -0.2836810 -0.4253238 -0.4525517 -0.2901264 -0.4671761
##            p_ert_cel     p_telj   p_boldog   p_egeszs  p_neg_erz   p_magany
## RC1        0.6604940  0.6053890  0.6577439  0.4361927 -0.5526917 -0.3836024
## RC2        0.2450133  0.2457362  0.3347994  0.2566204 -0.4805750 -0.2836810
## RC3        0.6050290  0.5759880  0.6490360  0.4511676 -0.6201278 -0.4253238
## p_poz_erz  0.7531407  0.7229889  0.8331380  0.5274025 -0.6866772 -0.4525517
## p_elmely   0.6305894  0.5485736  0.5095568  0.3379240 -0.4282924 -0.2901264
## p_poz_kapc 0.6495659  0.5369546  0.7279601  0.4356007 -0.4972611 -0.4671761
## p_ert_cel  1.0000000  0.7500587  0.7304123  0.4943210 -0.5041553 -0.3912630
## p_telj     0.7500587  1.0000000  0.6500552  0.5273559 -0.5219503 -0.3331066
## p_boldog   0.7304123  0.6500552  1.0000000  0.4527424 -0.6081244 -0.5236918
## p_egeszs   0.4943210  0.5273559  0.4527424  1.0000000 -0.4147676 -0.2980372
## p_neg_erz -0.5041553 -0.5219503 -0.6081244 -0.4147676  1.0000000  0.5487226
## p_magany  -0.3912630 -0.3331066 -0.5236918 -0.2980372  0.5487226  1.0000000
```

Interestingly there is a strong correlation between almost all of the variables.The strongest correlation is between RC1 and and the PERMA items is with *p_poz_erz* (r = 0.69)