Factor Analysis

2022/05/10

Psychometrics

Many of the things we want to measure are *constructs* that are not *directly* measurable. e.g. IQ, anxiety, risk

"A group of blind men heard that a strange animal, called an elephant, had been brought to the town, but none of them were aware of its shape and form. Out of curiosity, they said: "We must inspect and know it by touch, of which we are capable"."

Psychometrics

Many of the things we want to measure are *constructs* that are not *directly* measurable. e.g. IQ, anxiety, risk

We can try to capture different *aspects* of latent variables.

For example, we might ask a variety of different questions as with standard scales and questionnaires like

- HEXACO
- Historical Clinical Risk Management-20 (HCR-20)
- Patient Health Questionnaire 9 (PHQ-9)

The HEXACO personality measures

The HEXACO scale measures personality using 60 or 100 item questionnaires.

These questionnaires supposedly breaks personality down into six different factors:

- Honesty-Humility
- Emotionality
- eXtraversion [sic]
- Agreeableness
- Conscientiousness
- Openness to Experience

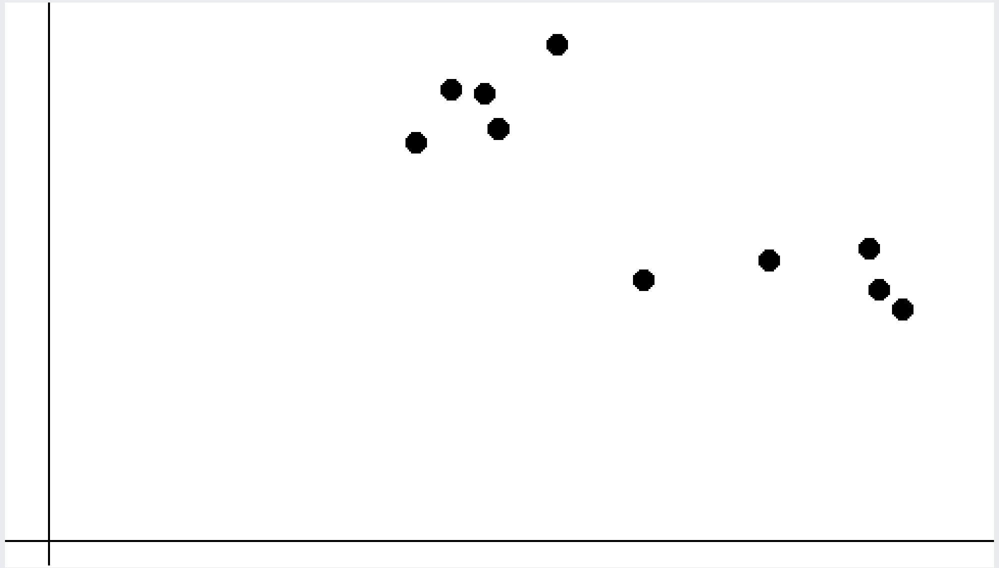
Example HEXACO items

1 = strongly disagree   2 = disagree   3 = neutral   4 = agree   5 = strongly agree

1	______	I would be quite bored by a visit to an art gallery.
2	______	I plan ahead and organize things, to avoid scrambling at the last minute.
3	______	I rarely hold a grudge, even against people who have badly wronged me.
4	______	I feel reasonably satisfied with myself overall.
5	______	I would feel afraid if I had to travel in bad weather conditions.
6	______	I wouldn't use flattery to get a raise or promotion at work, even if I thought it would succeed.
7	______	I'm interested in learning about the history and politics of other countries.
8	______	I often push myself very hard when trying to achieve a goal.
9	______	People sometimes tell me that I am too critical of others.
10	______	I rarely express my opinions in group meetings.

Performing factor and component analysis

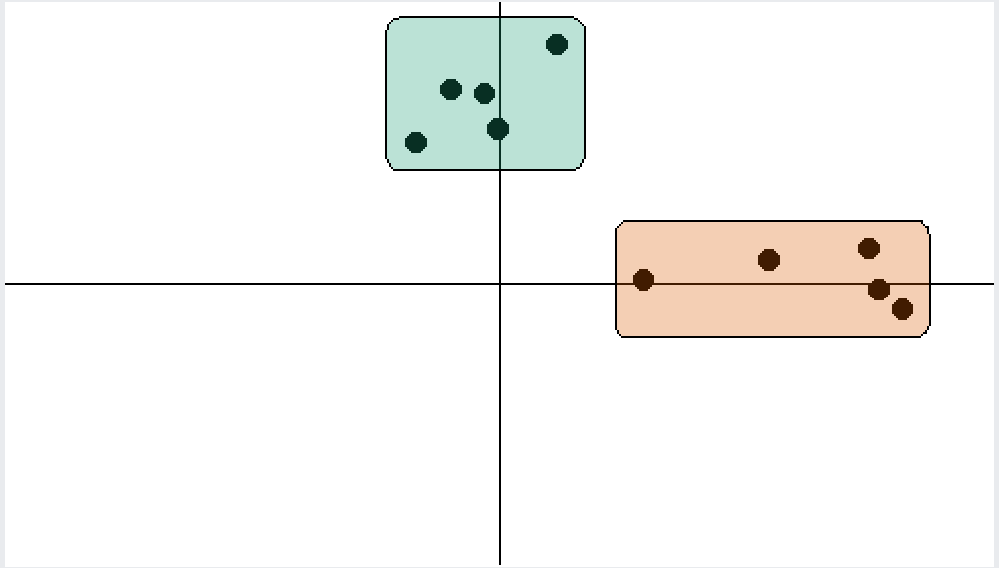
Graphical representation of factor analysis



Each axis is a *dimension* relating to an underlying construct.

In this example, based on the HEXACO scale, the x-axis represents the *Honesty-Humility* dimension, while the y-axis represents the *Emotionality* dimension.

Graphical representation of factor analysis



Each dot represents the score on an individual item.

The points that cluster together are correlated and are measuring part of the same underlying dimension.

We can shift the *axes* to pass through these points.

Factor loadings

Items that measure the *Emotionality* factor cluster - or **load** - high on the y-axis.

Items that measure the *Honesty-Humility* factor load high on the x-axis.

Factor loadings

The distance of an item from zero on a particular dimension indicates how heavily the item *loads* on that dimension.

Items that measure the *Emotionality* factor cluster - or **load** - high on the y-axis.

Items that measure the *Honesty-Humility* factor load high on the x-axis.

Factor loadings

The items that load on the Honesty-Humility axis are close to the centre of the *y-axis*, but distant from zero on the *x-axis*.

Factor loadings

The items that load on the Honesty-Humility axis are close to the centre of the *y-axis*, but distant from zero on the *x-axis*.

The items that load on the Emotionality factor are close to the centre of the *x-axis*, but distant from zero on the *y-axis*.

Factor loadings

Let's add a third set of items, a set of items that correlate with each other but not with either existing cluster.

These clearly load negatively on both our existing factors, but we may need another factor to characterise them properly.

Factor loadings

Let's add a third set of items, a set of items that correlate with each other but not with either existing cluster.

These clearly load negatively on both our existing factors, but we may need another factor to characterise them properly.

For each distinct *factor*, we need an additional *dimension*.

Preparing for factor analysis

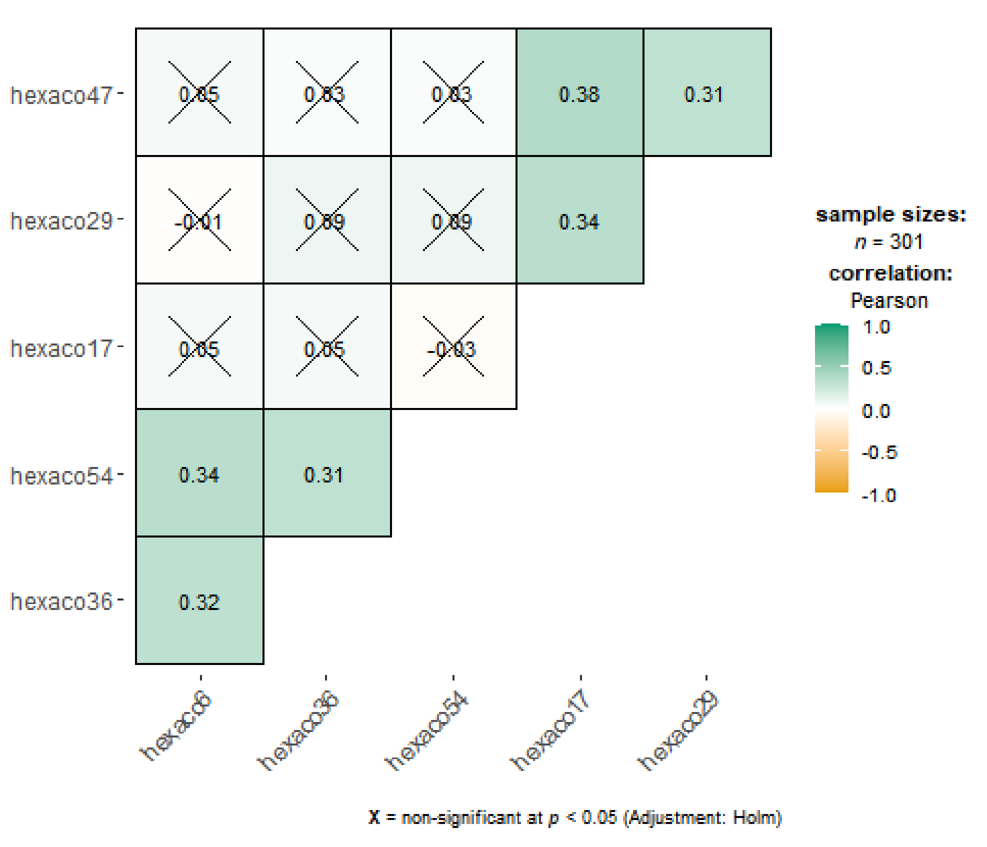
The *R-matrix*

Correlations are at the heart of how we understand which of our questionnaire items measure the same *factors*.

There are 60-item and 100-item versions of the HEXACO.

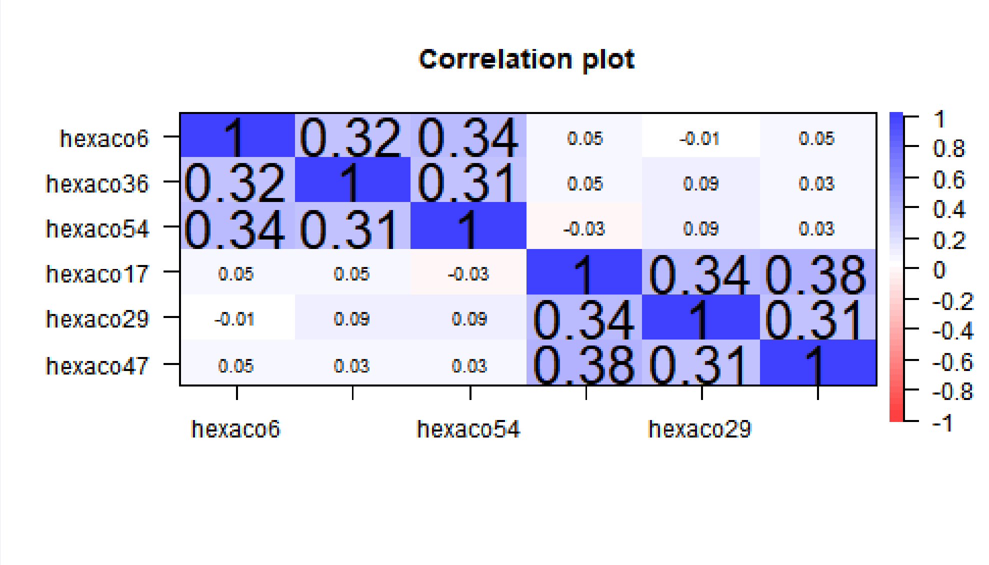
Here we take a look a small subset of those items.

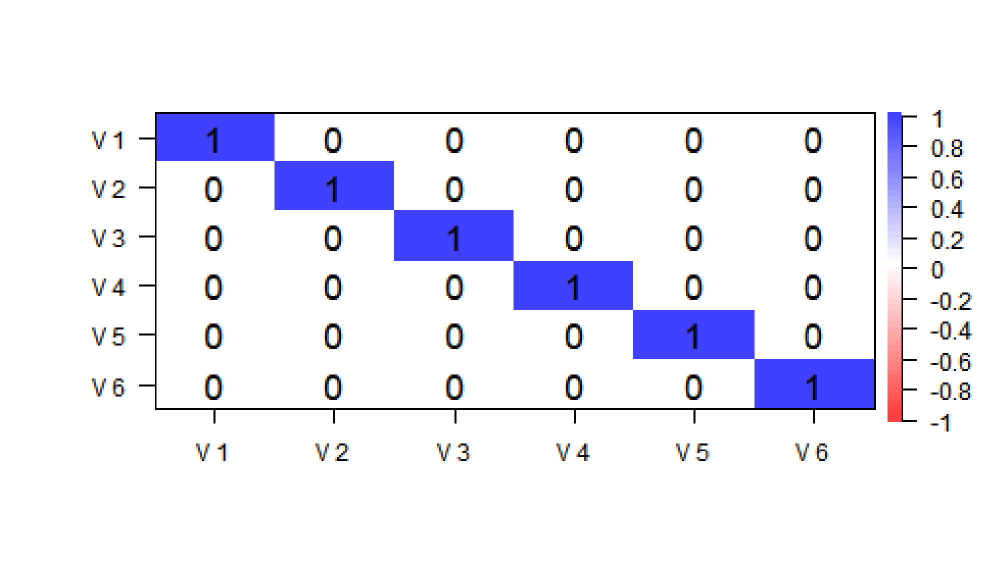
We have *two clusters* of items that correlate with *with each other* but not with the items in the *other* cluster.



The identity matrix

The matrix on the right is the *identity* matrix - this is what the correlation matrix would be like without structure.





Checking the *R-matrix*

We need to know whether there is sufficient correlative structure in the data!

The *Bartlett* test - run using the `check_sphericity_bartlett()` function from `parameters` - is used to check whether the correlation matrix significantly differs from the *identity* matrix.

```
check_sphericity_bartlett(hexaco_subset)
```

```
## # Test of Sphericity
##
## Bartlett's test of sphericity suggests that there is sufficient significant correlation
in the data for factor analysis (Chisq(15) = 187.45, p < .001).
```

Checking sampling adequacy

We also need to know if there is enough variability in the data.

The Kaiser-Meyer-Olkin statistic measures the degree to which each variable in the data can be predicted from the other variables.

KMO ranges from 0 to 1; values above .7 are generally considered acceptable.

```
check_kmo(hexaco_only)
```

```
## # KMO Measure of Sampling Adequacy
##
## The Kaiser, Meyer, Olkin (KMO) measure of sampling adequacy suggests that data seems
appropriate for factor analysis (KMO = 0.77).
```

Checking for sufficient factor structure

The `check_factorstructure()` function from `parameters` does both of these at once!

```
check_factorstructure(hexaco_only)
```

```
## # Is the data suitable for Factor Analysis?
##
##   - KMO: The Kaiser, Meyer, Olkin (KMO) measure of sampling adequacy suggests that data seems appropriate for factor analysis (KMO = 0.77).
##   - Sphericity: Bartlett's test of sphericity suggests that there is sufficient significant correlation in the data for factor analysis (Chisq(1770) = 7153.57, p < .001).
```

How many factors do we need?

How many factors do we need?

We need to figure out how many factors we need to break down our data.

In theory, we could have one per item.

... but that would be a lot of factors.

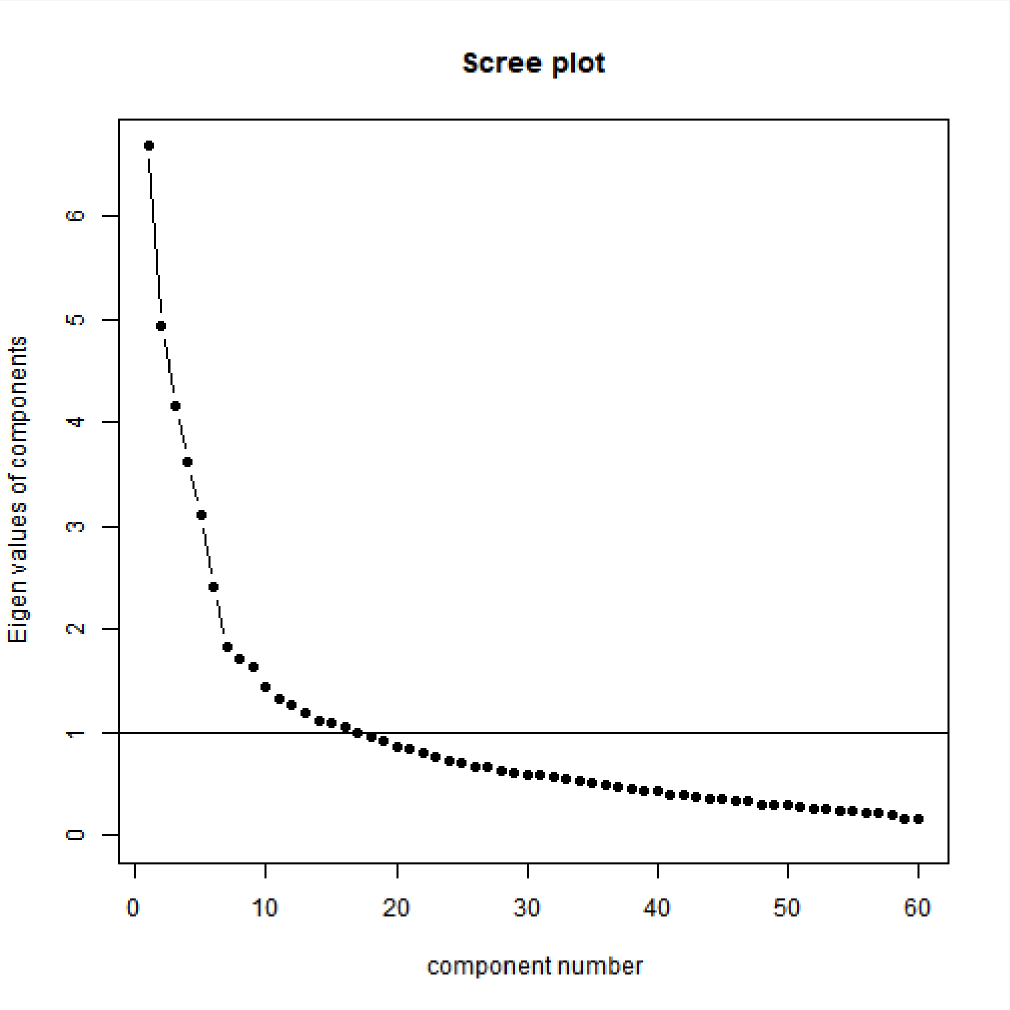
Here, it looks like there are at least five different groups.

Scree plots

Catell (1966) proposed the scree plot as a way to choose how many factors to keep.

The y-axis shows the eigenvalue of each potential factor, up to the maximum number possible.

```
scree(hexaco_only, factors = FALSE)
```



Eigenvalues

Eigenvalues tell us how much variance a particular factor explains.

Higher values mean more variance explained, and the more variance a factor explains, the more important it is.

They help us determine whether a factor is worth *extracting* for further analysis.

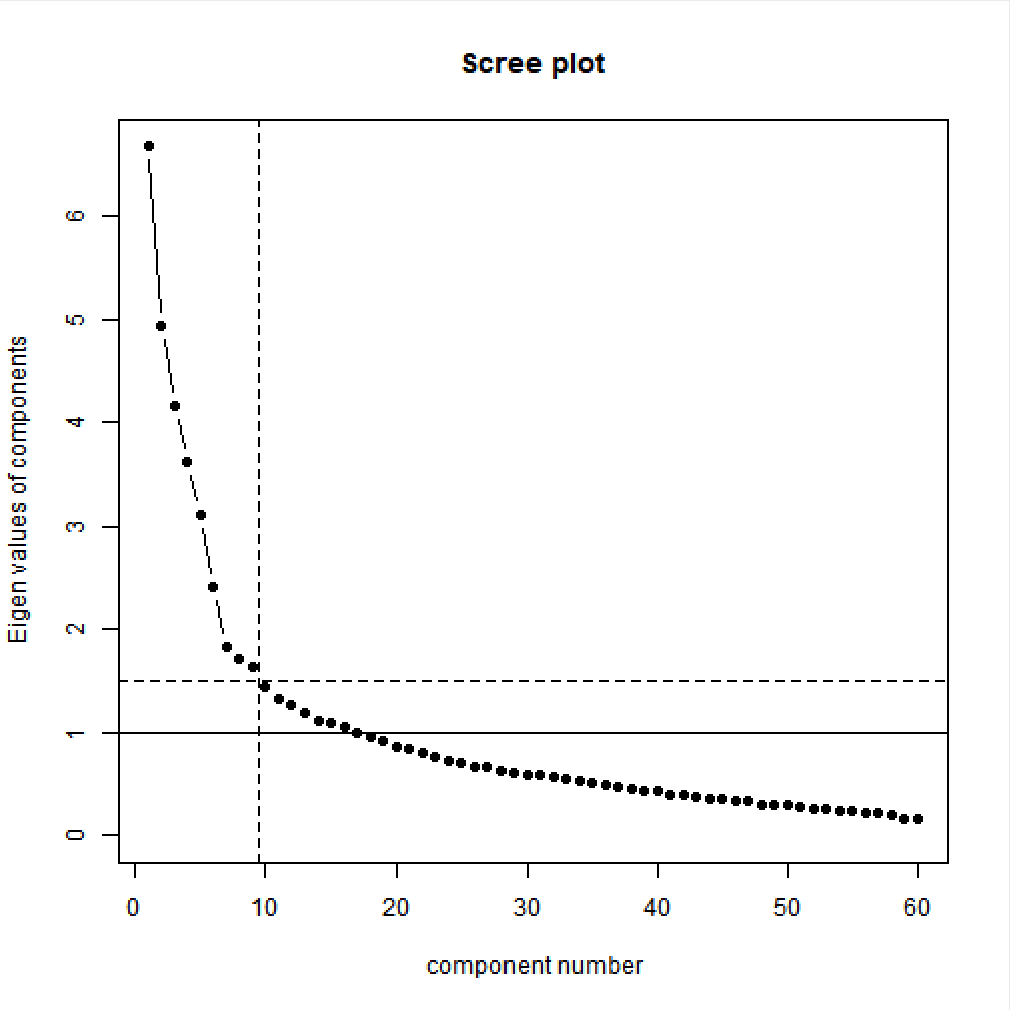
```
eigen(cor(hexaco_only))$values
```

```
##  [1] 6.6794022 4.9331351 4.1642486 3.6237441 3.1055095 2.4175969
##  [7] 1.8303788 1.7109257 1.6276844 1.4397559 1.3272585 1.2604329
## [13] 1.1922544 1.1163815 1.0921884 1.0615500 0.9873230 0.9548026
## [19] 0.9119362 0.8633100 0.8313116 0.7990644 0.7715206 0.7308254
## [25] 0.7084408 0.6727497 0.6619408 0.6319305 0.6145891 0.5917170
## [31] 0.5852278 0.5682678 0.5418668 0.5317604 0.5112533 0.4972359
## [37] 0.4697304 0.4557374 0.4375651 0.4280798 0.4018966 0.3945424
## [43] 0.3844344 0.3547073 0.3484113 0.3357983 0.3289566 0.3065944
## [49] 0.2977506 0.2890174 0.2725412 0.2599978 0.2595737 0.2436230
## [55] 0.2312924 0.2221693 0.2179546 0.1937423 0.1609424 0.1554209
```

Scree plots

We look for the *point of inflexion* - the point at which the eigenvalues have (more or less) stopped decreasing much.

It's probably around 9 components here!



Kaiser's criterion

An alternative to looking for the point of inflexion is to keep any factor where the eigenvalue is higher than 1 - this is called *Kaiser's criterion*.

This would pick out around *16* factors here.

Kaiser's criterion tends to keep too many factors.

Parallel analysis

Arguably the best method is **Parallel Analysis**, using the **fa.parallel()** function.

In parallel analysis, *random* data is generated and compared to the *true* data.

Factors above the *red* line should be kept. Here, it's 9, just like our "point of inflexion" rule would suggest.

```
fa.parallel(hexaco_only,
            fa = "pc")
```

```
## Parallel analysis suggests that the
number of factors =  NA  and the number of
components =  9
```

Principal Component Analysis

Principal Component Analysis

There are a number of differenct factor analysis methods available. We'll look at PCA. PCA is a *dimension reduction* method.

It produces a *simplified model* of the data that captures the inter-relationships between variables.

To run PCA on this kind of data, we can use the **principal()** function from the **psych** package.

```
?principal
```

Principal Component Analysis

Having decided we need nine factors, we use the **principal()** function to extract them from the data.

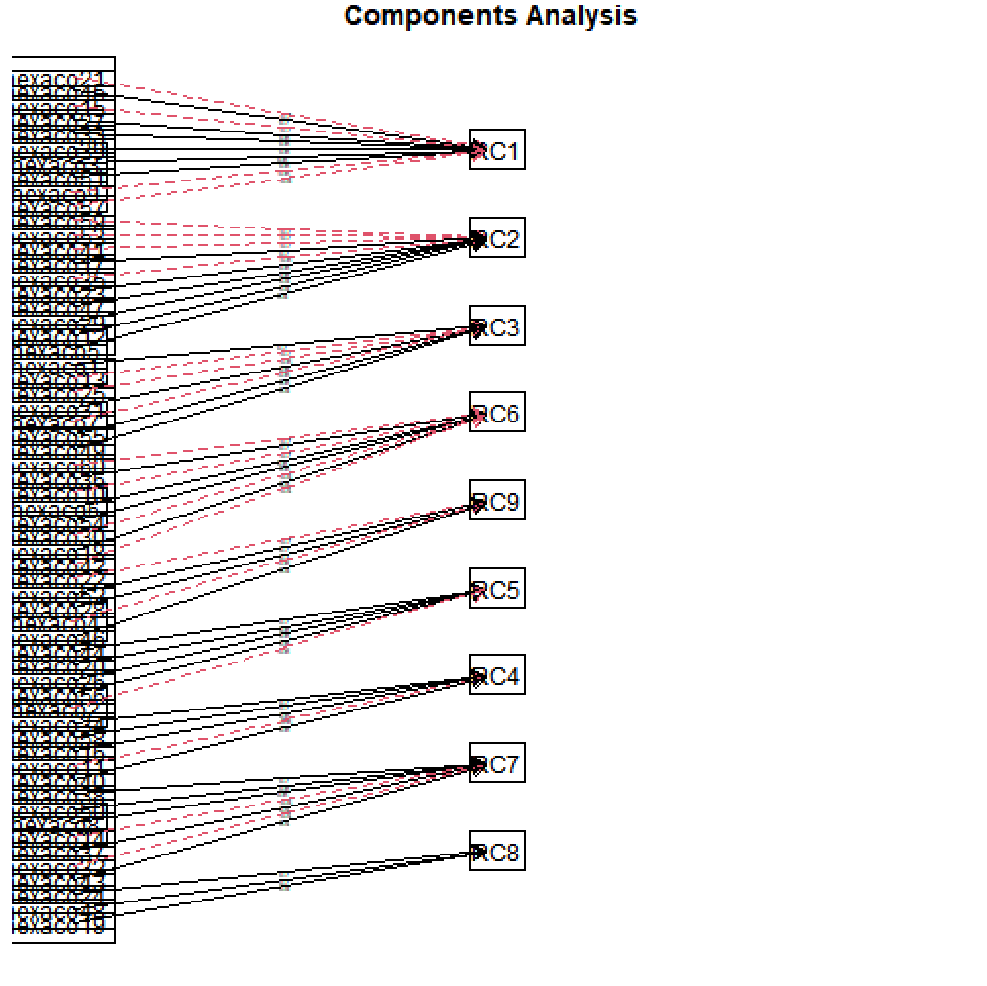
```
pca_hexa <- principal(hexaco_only,
                      nfactors = 9,
                      rotate = "varimax")
```

(Output is on the next slide!)

```r
pca_hexa
```

```
## Principal Components Analysis
## Call: principal(r = hexaco_only, nfactors = 9, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##             RC1   RC2   RC3   RC6   RC9   RC5   RC4   RC7   RC8   h2
## hexaco1  -0.17  0.00  0.74  0.03 -0.01 -0.03  0.09  0.04  0.05 0.59
## hexaco2  -0.01  0.21  0.02  0.10 -0.05 -0.58 -0.03  0.24  0.17 0.48
## hexaco3   0.59 -0.06 -0.09  0.19 -0.15  0.06  0.02  0.01  0.15 0.44
## hexaco4   0.15 -0.12  0.08  0.17 -0.59 -0.25  0.09  0.23  0.09 0.56
## hexaco5   0.10  0.50  0.03 -0.14 -0.06 -0.06 -0.08  0.03  0.01 0.30
## hexaco6   0.14  0.02  0.06  0.64  0.06 -0.03  0.00  0.06 -0.24 0.49
## hexaco7  -0.01 -0.06 -0.59  0.04  0.06 -0.14  0.07 -0.03 -0.08 0.39
## hexaco8  -0.01  0.10 -0.03  0.13 -0.23 -0.19  0.20  0.61  0.06 0.53
## hexaco9  -0.58 -0.10 -0.12 -0.12  0.08  0.00 -0.03  0.07  0.19 0.42
## hexaco10 -0.11 -0.27  0.09 -0.66  0.15  0.21  0.02  0.08 -0.03 0.60
## hexaco11  0.15  0.16  0.32 -0.12  0.02  0.14 -0.64  0.12  0.01 0.60
## hexaco12 -0.05  0.53  0.00  0.02  0.37 -0.05 -0.07  0.26  0.07 0.50
## hexaco13  0.00  0.07 -0.65  0.05 -0.01  0.36 -0.14  0.21 -0.08 0.63
## hexaco14 -0.11 -0.12  0.19 -0.06 -0.05  0.30  0.01 -0.48  0.20 0.43
## hexaco15 -0.65 -0.01  0.01  0.00  0.10  0.14 -0.05  0.24  0.07 0.52
## hexaco16  0.09  0.16  0.21  0.01 -0.21  0.14  0.65  0.06 -0.13 0.58
## hexaco17  0.14  0.67  0.04 -0.03  0.09  0.03  0.15  0.18  0.00 0.54
## hexaco18  0.22  0.03 -0.05  0.46  0.11  0.08  0.08  0.17  0.08 0.33
## hexaco19 -0.11  0.01  0.33  0.11 -0.19  0.05 -0.21 -0.11  0.47 0.45
## hexaco20  0.05 -0.12  0.27 -0.16  0.08  0.64  0.05 -0.15  0.13 0.57
```

```
fa.diagram(pca_hexa)
```



Factor Rotation

Factor rotation

Factor rotation

Factor rotation

Orthogonal rotation

Orthogonal rotation

Orthogonal rotation

The typical method of orthogonal rotation is called *varimax*.

```r
principal(hexaco_only,
          nfactors = 9,
          rotate = "varimax")
```

```
## Principal Components Analysis
## Call: principal(r = hexaco_only, nfactors = 9, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##             RC1   RC2   RC3   RC6   RC9   RC5   RC4   RC7   RC8   h2
## hexaco1   -0.17  0.00  0.74  0.03 -0.01 -0.03  0.09  0.04  0.05 0.59
## hexaco2   -0.01  0.21  0.02  0.10 -0.05 -0.58 -0.03  0.24  0.17 0.48
## hexaco3    0.59 -0.06 -0.09  0.19 -0.15  0.06  0.02  0.01  0.15 0.44
## hexaco4    0.15 -0.12  0.08  0.17 -0.59 -0.25  0.09  0.23  0.09 0.56
## hexaco5    0.10  0.50  0.03 -0.14 -0.06 -0.06 -0.08  0.03  0.01 0.30
## hexaco6    0.14  0.02  0.06  0.64  0.06 -0.03  0.00  0.06 -0.24 0.49
## hexaco7   -0.01 -0.06 -0.59  0.04  0.06 -0.14  0.07 -0.03 -0.08 0.39
## hexaco8   -0.01  0.10 -0.03  0.13 -0.23 -0.19  0.20  0.61  0.06 0.53
## hexaco9   -0.58 -0.10 -0.12 -0.12  0.08  0.00 -0.03  0.07  0.19 0.42
## hexaco10  -0.11 -0.27  0.09 -0.66  0.15  0.21  0.02  0.08 -0.03 0.60
```

Oblique rotation

Oblique rotation

Oblique rotation

The typical method of oblique rotation is called *oblimin*.

```
principal(hexaco_only,
          nfactors = 9,
          rotate = "oblimin")
```

```
## Loading required namespace: GPArotation

## Principal Components Analysis
## Call: principal(r = hexaco_only, nfactors = 9, rotate = "oblimin")
## Standardized loadings (pattern matrix) based upon correlation matrix
##             TC1   TC2   TC3   TC6   TC5   TC4   TC7   TC9   TC8   h2
## hexaco1   -0.18 -0.03  0.74  0.07  0.10 -0.05  0.02  0.15  0.03 0.59
## hexaco2   -0.01  0.16  0.10  0.12 -0.55 -0.03 -0.04  0.21  0.20 0.48
## hexaco3    0.57 -0.08 -0.10  0.17  0.07 -0.12 -0.01 -0.02  0.15 0.44
## hexaco4    0.08 -0.13  0.06  0.17 -0.21 -0.58  0.03  0.20  0.08 0.56
## hexaco5    0.09  0.50 -0.02 -0.15 -0.05 -0.08 -0.09  0.01  0.02 0.30
## hexaco6    0.10 -0.05  0.10  0.59  0.03  0.11  0.01  0.06 -0.28 0.49
## hexaco7    0.01 -0.04 -0.52  0.02 -0.24  0.11  0.14 -0.11 -0.06 0.39
## hexaco8   -0.04  0.03 -0.01  0.10 -0.17 -0.19  0.16  0.59  0.05 0.53
```

Which rotation to use?

For the most part, use *orthogonal* rotation (i.e. Varimax).

Oblique rotation is defensible when there are *a priori*, *theoretical* reasons to believe there will be correlations between dimensions.

```
principal(hexaco_only,
          nfactors = 9,
          rotate = "varimax")
```

```
principal(hexaco_only,
          nfactors = 9,
          rotate = "oblimin")
```

Factor interpretation

Final PCA

Let's finish off by looking closely at the PCA solution with nine factors and *varimax* rotation.

```
pca_hexa
```

```
## Principal Components Analysis
## Call: principal(r = hexaco_only, nfactors = 9, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##             RC1   RC2   RC3   RC6   RC9   RC5   RC4   RC7   RC8   h2
## hexaco1   -0.17  0.00  0.74  0.03 -0.01 -0.03  0.09  0.04  0.05 0.59
## hexaco2   -0.01  0.21  0.02  0.10 -0.05 -0.58 -0.03  0.24  0.17 0.48
## hexaco3    0.59 -0.06 -0.09  0.19 -0.15  0.06  0.02  0.01  0.15 0.44
## hexaco4    0.15 -0.12  0.08  0.17 -0.59 -0.25  0.09  0.23  0.09 0.56
## hexaco5    0.10  0.50  0.03 -0.14 -0.06 -0.06 -0.08  0.03  0.01 0.30
## hexaco6    0.14  0.02  0.06  0.64  0.06 -0.03  0.00  0.06 -0.24 0.49
## hexaco7   -0.01 -0.06 -0.59  0.04  0.06 -0.14  0.07 -0.03 -0.08 0.39
## hexaco8   -0.01  0.10 -0.03  0.13 -0.23 -0.19  0.20  0.61  0.06 0.53
## hexaco9   -0.58 -0.10 -0.12 -0.12  0.08  0.00 -0.03  0.07  0.19 0.42
## hexaco10 -0.11 -0.27  0.09 -0.66  0.15  0.21  0.02  0.08 -0.03 0.60
## hexaco11  0.15  0.16  0.32 -0.12  0.02  0.14 -0.64  0.12  0.01 0.60
## hexaco12 -0.05  0.53  0.00  0.02  0.37 -0.05 -0.07  0.26  0.07 0.50
```

Final PCA

Down at the bottom of our output are statistics about the amount of variance our factors explain.

```
##                               RC1        RC2        RC3        RC6
## SS loadings           4.4744097 4.45609263 3.54088899 3.46245860
## Proportion Var        0.0745735 0.07426821 0.05901482 0.05770764
## Cumulative Var        0.0745735 0.14884171 0.20785652 0.26556417
## Proportion Explained  0.1486879 0.14807922 0.11766634 0.11506004
## Cumulative Proportion 0.1486879 0.29676714 0.41443348 0.52949352
##                              RC9        RC5        RC4        RC7
## SS loadings           3.18062363 3.14794568 3.01616898 2.87416485
## Proportion Var        0.05301039 0.05246576 0.05026948 0.04790275
## Cumulative Var        0.31857456 0.37104032 0.42130980 0.46921255
## Proportion Explained  0.10569446 0.10460854 0.10022951 0.09551061
## Cumulative Proportion 0.63518797 0.73979651 0.84002602 0.93553663
##                              RC8
## SS loadings           1.93987219
## Proportion Var        0.03233120
## Cumulative Var        0.50154376
## Proportion Explained  0.06446337
## Cumulative Proportion 1.00000000
```

Interpreting the output

It looks like there are 10 items that load on our first factor.

The top three are the following items from the HEXACO-60:

Item 21: People think of me as someone who has a quick temper.

Item 45: Most people tend to get angry more quickly than I do.

Item 15: People sometimes tell me that I'm too stubborn.

Interpreting the output

In fact, the ten items are all those that correspond to *Agreeableness*:

Agreeableness	
Forgiveness	3, 27
Gentleness	9R, 33, 51
Flexibility	15R, 39, 57R
Patience	21R, 45

Note that several should be reversed, and they have *negative* factor loadings because we didn't actually reverse them!

How do individual *participants* score?

Once we know what our *factors* are, how do we convert each participant's data into something that tells us how that participant rated for each factor?

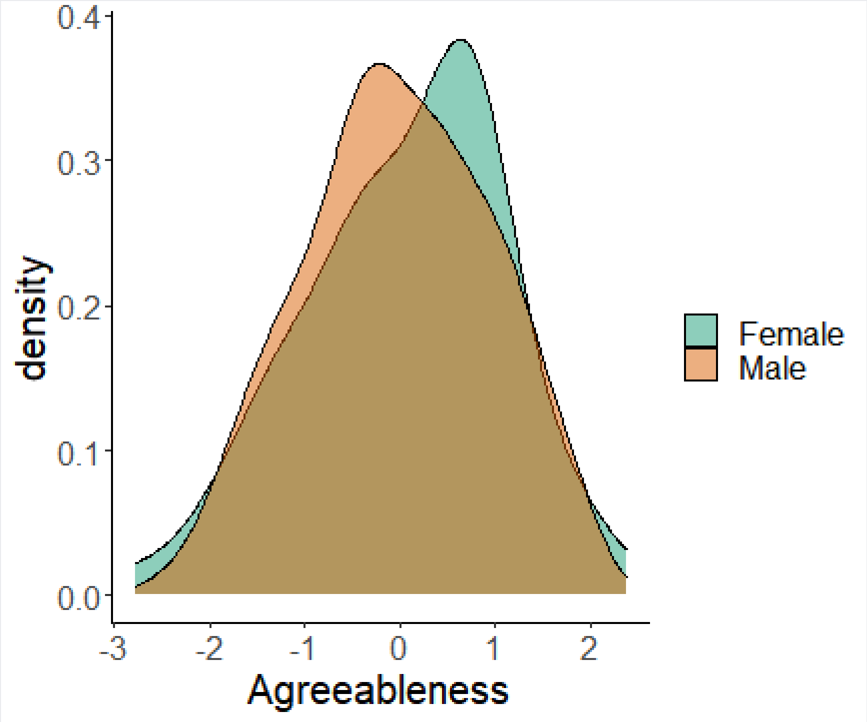
```r
head(pca_hexa$scores)
```

```
##             RC1         RC2        RC3        RC6        RC9
## [1,] -2.0370952 -0.74766949  1.0906981  2.4308437 -0.7581765
## [2,] -1.3067772  0.07377406 -0.9048852  0.6698510 -1.3186814
## [3,] -0.4223559 -0.43381267 -1.0134896 -0.6835562  1.2656719
## [4,] -1.4688597 -2.01939403 -1.6466062  2.3991087  0.1195594
## [5,]  0.8065396  0.42209968 -0.6753594  0.9384934 -1.7880986
## [6,]  0.4092487 -1.41218241 -1.5163114 -2.2545879  1.9586546
##             RC5         RC4        RC7        RC8
## [1,] -0.5199759 -0.64786941  1.4417465 -0.2530221
## [2,] -0.4670667 -0.49776904 -0.2658352 -0.8120384
## [3,] -1.1032942  0.19409792  0.6168094  0.2996072
## [4,] -1.3225573 -1.10714105 -0.1981588 -0.4550324
## [5,] -0.7686581  1.06179528 -1.2344637  0.4208103
## [6,] -1.7329438  0.03392348 -1.0618261 -1.5631121
```

A quick example

The factor scores can be treated as if they were any other variable! Here I combine the Factor Scores with the original data.

```
final_data <- cbind(crime,
                    pca_hexa$scores)
ggplot(final_data,
       aes(x = RC1,
           fill = factor(sex,
                         levels = c(1,
2),
                         labels =
c("Female", "Male")))) +
  geom_density(alpha = 0.5) +
  scale_fill_brewer(palette = "Dark2") +
  labs(x = "Agreeableness",
       fill = "") +
  theme_classic() +
  theme(text = element_text(size = 20))
```



Why would you do this?

Why use factor analysis?

1) Rather than trying to analyse many, many different items as if they are each independent from each other, you can reduce the task down to a smaller set of factors

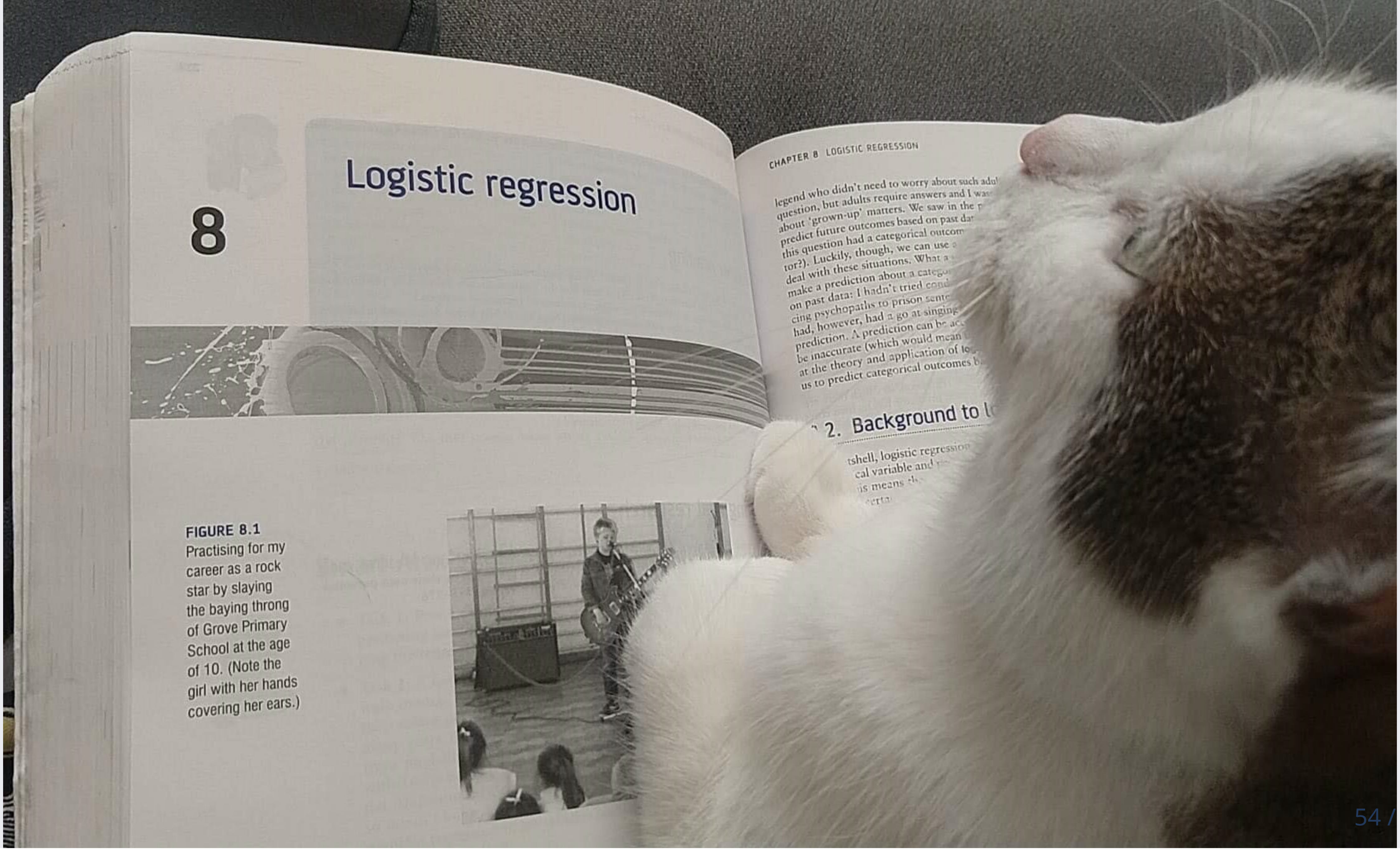
2) Factor analysis helps you *condense* the information down, while still retaining the benefit of having many different, independent measurements of the underlying constructs.

3) During the *design* of questionnaires, it helps you work out which items are measuring which thing, and which items are worth keeping!

This week's background

Background reading for this week can be found in Field et al, Discovering Statistics Using R (2011), Chapter 17 - Exploratory Factor Analysis.

There is a Datacamp course, Factor Analysis in R. Note: it's a little tough in places - don't be discouraged! It's good practice and covers some topics we didn't cover today!


8
Logistic regression
CHAPTER 8 LOGISTIC REGRESSION
FIGURE 8.1
Practising for my career as a rock star by slaying the baying throng of Grove Primary School at the age of 10. (Note the girl with her hands covering her ears.)