Item Analysis Example

Louis Rocconi
MAD with Methods and Measures Blog
2023

```
################################
####  Item Analysis Example ####
################################

# Load packages
library(psych)

# Load data
saq6 <- read.csv(url("https://lrocconi.github.io/files/saq6.csv"))

# These data are a subset of items from Andy Fields' SPSS Anxiety Questionnaire.

# Response options: Strongly Disagree, Disagree, Neutral, Agree, Strongly Agree

# stat_cry ==   Statistics make me cry.
# sd_excite ==  Standard deviations excite me.
# corr_attack ==    I dream that Pearson is attacking me with correlation coefficients.
# understand_stat == I don’t understand statistics.
# sleep_eign == I can’t sleep for thoughts of eigenvectors.
# duvet_normal ==   I wake up under my duvet thinking that I am trapped under a normal
distribution.

# The alpha function from the psych package is a great way to start and computes
# many of the item analysis statistics we discussed.
psych::alpha(saq6)
```

```
## Warning in psych::alpha(saq6): Some items were negatively correlated with the first
principal component and probably
## should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' option
```

```
## Some items ( sd_excite ) were negatively correlated with the first principal component
and
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' option

##
## Reliability analysis
## Call: psych::alpha(x = saq6)
##
##   raw_alpha std.alpha G6(smc) average_r  S/N   ase mean   sd median_r
##       0.37      0.43    0.54      0.11 0.75 0.018  2.9 0.48     0.24
##
##     95% confidence boundaries
##          lower alpha upper
## Feldt     0.33  0.37  0.41
## Duhachek  0.34  0.37  0.41
```

```
## 
##  Reliability if an item is dropped:
##                 raw_alpha std.alpha G6(smc) average_r  S/N alpha se  var.r
## stat_cry             0.20      0.25    0.42     0.062 0.33   0.0232 0.1378
## sd_excite            0.72      0.72    0.70     0.344 2.62   0.0088 0.0092
## corr_attack          0.16      0.23    0.39     0.056 0.30   0.0247 0.1272
## understand_stat      0.18      0.25    0.43     0.064 0.34   0.0238 0.1430
## sleep_eign           0.24      0.32    0.46     0.087 0.48   0.0223 0.1505
## duvet_normal         0.13      0.22    0.37     0.054 0.29   0.0257 0.1208
##                 med.r
## stat_cry         0.22
## sd_excite        0.37
## corr_attack      0.21
## understand_stat  0.23
## sleep_eign       0.33
## duvet_normal     0.21
## 
##  Item statistics 
##                    n raw.r std.r r.cor r.drop mean   sd
## stat_cry        2571  0.63  0.67  0.59   0.40  2.4 0.83
## sd_excite       2571 -0.21 -0.25 -0.67  -0.51  2.6 1.08
## corr_attack     2571  0.67  0.69  0.64   0.42  2.8 0.95
## understand_stat 2571  0.65  0.66  0.57   0.39  2.7 0.96
## sleep_eign      2571  0.61  0.59  0.46   0.30  3.6 1.04
## duvet_normal    2571  0.70  0.70  0.66   0.44  3.2 0.98
## 
## Non missing response frequency for each item
##                    1    2    3    4    5 miss
## stat_cry        0.11 0.52 0.29 0.07 0.02    0
## sd_excite       0.19 0.26 0.34 0.17 0.03    0
## corr_attack     0.05 0.37 0.36 0.17 0.05    0
## understand_stat 0.06 0.43 0.29 0.18 0.04    0
## sleep_eign      0.02 0.15 0.25 0.37 0.22    0
## duvet_normal    0.02 0.26 0.34 0.29 0.09    0

# Notice the warning message and the low alpha (0.37).  This indicates an item
# may need to be reverse coded. We can use the item discrimination to figure our
# which one! In the Item Statistics table, look at the r.drop column. Notice
# that sd_excite has a negative discrimination value. In this case, this item
# should be revise coded, since lower responses on the item "Standard deviations
# excite me" indicate more anxiety.

# Let's reverse code the item and try it again.
table(saq6$sd_excite)

## 
##   1   2   3   4   5 
## 497 672 878 448  76

saq6$sd_excite <- 6 - saq6$sd_excite

# Let's check to see if the reverse coding worked.
table(saq6$sd_excite)
```

```
##
##   1   2   3   4   5
##  76 448 878 672 497

# reverse coding worked!

# Let's re-run alpha
psych::alpha(saq6)

##
## Reliability analysis
## Call: psych::alpha(x = saq6)
##
##   raw_alpha std.alpha G6(smc) average_r S/N    ase mean   sd median_r
##       0.76      0.76    0.74      0.35 3.2 0.0074    3 0.66     0.34
##
##     95% confidence boundaries
##          lower alpha upper
## Feldt     0.74  0.76  0.77
## Duhachek  0.74  0.76  0.77
##
##  Reliability if an item is dropped:
##                  raw_alpha std.alpha G6(smc) average_r S/N alpha se  var.r med.r
## stat_cry             0.73      0.73    0.69      0.35 2.7   0.0085 0.0069  0.36
## sd_excite            0.72      0.72    0.70      0.34 2.6   0.0088 0.0092  0.37
## corr_attack          0.71      0.71    0.68      0.33 2.5   0.0089 0.0070  0.33
## understand_stat      0.73      0.73    0.70      0.36 2.8   0.0083 0.0068  0.36
## sleep_eign           0.75      0.75    0.71      0.38 3.0   0.0079 0.0020  0.39
## duvet_normal         0.70      0.71    0.67      0.32 2.4   0.0093 0.0069  0.33
##
##  Item statistics
##                    n raw.r std.r r.cor r.drop mean   sd
## stat_cry        2571  0.64  0.67  0.57   0.49  2.4 0.83
## sd_excite       2571  0.70  0.68  0.58   0.51  3.4 1.08
## corr_attack     2571  0.70  0.71  0.63   0.54  2.8 0.95
## understand_stat 2571  0.65  0.65  0.55   0.47  2.7 0.96
## sleep_eign      2571  0.62  0.60  0.48   0.41  3.6 1.04
## duvet_normal    2571  0.74  0.73  0.67   0.58  3.2 0.98
##
## Non missing response frequency for each item
##                    1    2    3    4    5 miss
## stat_cry        0.11 0.52 0.29 0.07 0.02    0
## sd_excite       0.03 0.17 0.34 0.26 0.19    0
## corr_attack     0.05 0.37 0.36 0.17 0.05    0
## understand_stat 0.06 0.43 0.29 0.18 0.04    0
## sleep_eign      0.02 0.15 0.25 0.37 0.22    0
## duvet_normal    0.02 0.26 0.34 0.29 0.09    0

# The alpha is much better (.76) and all the discrimination statistics are
# positive and all within acceptable ranges (.3 to .7). The item duvet_normal
# has the largest item-total correlation (.58) which indicates it best taps into
# the latent construct or best differentiates between those with high and low
# anxiety. The item sleep_eigen has the lowest item-total correlation (but still
# acceptable). Item means (i.e., endorsability) and standard deviation (sd) are
# given in the item statistics table. Item mean indicates location on the latent
# construct and sd provides a measure of respondent variability. We also have a
```

```r
# table of response frequency for each item, which indicate whether each
# response option was endorsed.

# We can divide the item means the maximum response option to convert the mean
# to a difficulty value ranging from 0 to 1
colMeans(saq6)/5
```

```
##        stat_cry       sd_excite     corr_attack understand_stat      sleep_eign
##       0.4748347       0.6829249       0.5572151       0.5444574       0.7248541
##    duvet_normal
##       0.6341501
```

```r
# Difficulty parameters are all within an acceptable range. The item sleep_eign
# is a on the high end, indicating a high endorsability for this item. Maybe
# this item isn't giving us as much information as the other items about
# anxiety.

# I like to examine discrmination coefficients for each response option, so I
# wrote a function to compute those for me.
item_distractor <- function(df) {
  # Compute total score for each item
  df$total_score <- rowSums(df)

  # Function to calculate point-biserial correlation
  point_biserial <- function(item, total_score) {
    levels <- sort(unique(item))
    correlations <- sapply(levels, function(level) {
      binary <- as.numeric(item == level)
      corrected_total_score <- total_score - as.numeric(as.character(item))
      cor(binary, corrected_total_score, method = "pearson")
          })
    names(correlations) <- levels
    correlations
  }

  # Apply the function to each item (excluding 'total_score')
  item_vars <- setdiff(names(df), "total_score")
  correlations <- sapply(df[item_vars], point_biserial, total_score = df$total_score)

  # Convert to data frame and set row and column names
  correlations_df <- as.data.frame(correlations)

  # Transpose the data frame
  results <- t(correlations_df)

  return(results)
}

item_distractor(saq6)
```

```
##                           1          2           3         4         5
## stat_cry        -0.22577308 -0.2804601  0.23400039 0.2988741 0.2074142
## sd_excite       -0.14882437 -0.3110136 -0.19483729 0.1978048 0.3764655
## corr_attack     -0.23456393 -0.3609829  0.10736561 0.3166838 0.2689526
## understand_stat -0.19173679 -0.3210727  0.11426665 0.2670030 0.2454841
## sleep_eign      -0.01452337 -0.2421584 -0.22810978 0.0648654 0.3741233
## duvet_normal    -0.11044303 -0.4292772 -0.09274881 0.3117854 0.3690101

# We want to see negative discrimination values for strongly disagree (1) and
# disagree (2), indicating those who respond in that category have lower total
# scores (or lower anxiety) and positive correlations for agree (4) and strongly
# agree (5) indicating that people choosing those categories have higher anxiety (as
# reflected in the total score). All of the items follow this patten; however,
# sleep_eign is interesting. There is essentially no correlation between the
# total score those who respond "Strongly disagree" and "Agree". However, very
# few respondents selected "Strongly disagree" for that item (2%) but the
# largest proportion choose "Agree" (37%). Given what all we found about this
# item, we may want to review the item and entertain revisions.
```