Assignment 6 – Reliability

Manual spot checks included to ensure correct data processing

The final reliability output is at the bottom

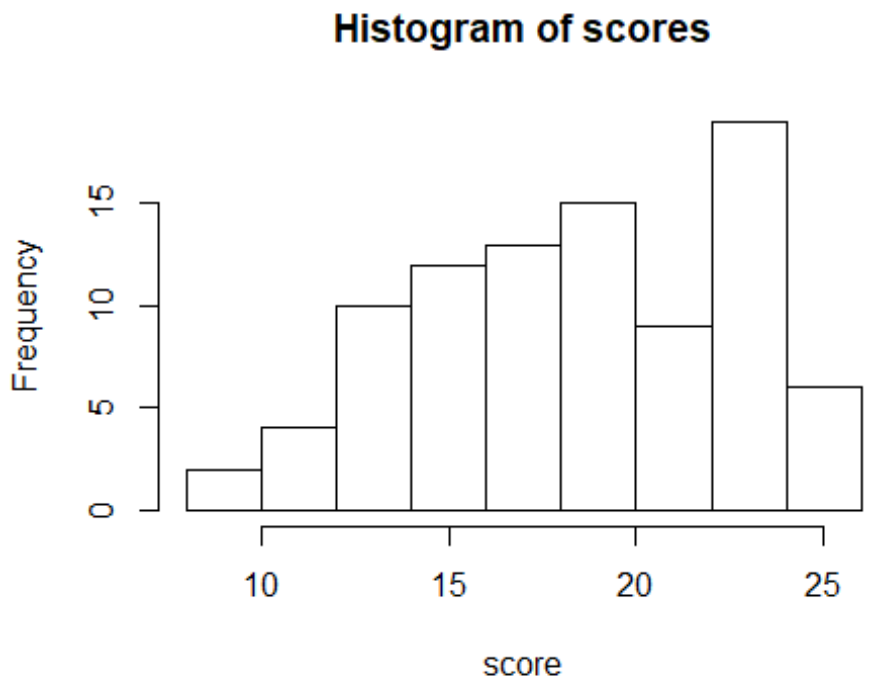
Stefan Maciolek

11/1/2019

```
#importing item data
library(foreign)
itemdata <- read.spss("D:\\515\\Week 9 10-28\\ItemDataHW.sav", to.data.frame
= TRUE)

## re-encoding from CP1252

#importing key
key <- read.table("D:\\515\\Week 9 10-28\\key.txt", header = TRUE)

#converting key to numeric
for(i in 1:29){
  if (key$Key[i] == "A") {key$newKey[i] = 1}
  else if (key$Key[i] == "B") {key$newKey[i] = 2}
  else if (key$Key[i] == "C") {key$newKey[i] = 3}
  else if (key$Key[i] == "D") {key$newKey[i] = 4}
}
#manual check, remove if no longer necessary
print("Spot check that the numbers in newKey correspond to the correct answer
s in Key")

## [1] "Spot check that the numbers in newKey correspond to the correct answe
rs in Key"

print(key[1:5,2:3])

##   Key newKey
## 1   A      1
## 2   A      1
## 3   C      3
## 4   C      3
## 5   D      4

#creating right/wrong matrix
correct <- rep(0,90*29)
dim(correct) <- c(90,29)
correct <- as.data.frame(correct)
for (j in 1:90) {
  for (i in 1:29){
    if (is.na(itemdata[j,i]) == TRUE) {itemdata[j,i]=0} #any omits are wrong
```

```
    if (key$newKey[i] == itemdata[j,i]) {correct[j,i]=1}  #checking item resp
onses against key
  }
}
#manual check, remove if no longer necessary
print("spot check that the each entry in correct = 1 if the itemdata entry ma
tches the key, and 0 otherwise")

## [1] "spot check that the each entry in correct = 1 if the itemdata entry m
atches the key, and 0 otherwise"

print(key[1:2,3])

## [1] 1 1

print(itemdata[1:5,1:2])

##   Q1 Q2
## 1  1  1
## 2  2  4
## 3  1  4
## 4  1  3
## 5  1  1

print(correct[1:5,1:2])

##   V1 V2
## 1  1  1
## 2  0  0
## 3  1  0
## 4  1  0
## 5  1  1

#computing scores for each examinee
score <- rep(0,90)
for (i in 1:90){
score[i] = sum(correct[i,1:29])
}
print("spot check that the scores seem reasonable and range from 0 to 29")

## [1] "spot check that the scores seem reasonable and range from 0 to 29"

print(score[1:10])

##  [1] 26 13 22 24 20 24 24 24 23  9

#computing the means and sds for each item
mean <- rep(0,29)
sd <- rep(0,29)
var <- rep(0,29)
for (i in 1:29){
  mean[i] <- mean(correct[1:90,i])
```

```
  sd[i] <- sd(correct[1:90,i])
}
var <- sd^2
meanscore <- mean(score)
sdscore <- sd(score)
print(paste("The mean of the scores is: ",round(meanscore,4)))

## [1] "The mean of the scores is:  18.8333"

print(paste("The standard deviation of the scores is ",round(sdscore,4)))

## [1] "The standard deviation of the scores is  4.2221"

print("The means and standard deviations of the items were also calculated bu
t are not displayed")

## [1] "The means and standard deviations of the items were also calculated b
ut are not displayed"

print("spot check that the item means and sds all seem reasonable and range f
rom 0 to 1")

## [1] "spot check that the item means and sds all seem reasonable and range
from 0 to 1"

print(round(mean[1:5],4))

## [1] 0.7000 0.4000 1.0000 0.8667 0.8444

print(round(sd[1:5],4))

## [1] 0.4608 0.4926 0.0000 0.3418 0.3645

hist(score, main = "Histogram of scores")
```




```
#computing reliability with Cronbach's Alpha
alpha <- (29/(29-1)) * (1-(sum(var[1:29])/(sdscore^2)))
print(paste("The reliability, calculated via Cronbach's alpha is: ",round(alp
ha,4)))
```

```
## [1] "The reliability, calculated via Cronbach's alpha is:  0.7197"
```

```
#item deletion reliabilities:
delete <- function(i){
  sdscoredelete <- sd(score[-i])
  deletedalpha <- (28/(28-1)) * (1-(sum(var[-i])/(sdscoredelete^2)))
  return(round(deletedalpha,4))
}
delete(1)
```

```
## [1] 0.7262
```

```
missingalpha <- rep(0,29)
for (i in 1:29){
  missingalpha[i] <- delete(i)
}
itemnumber <- c(1:29)
partialalphas <- cbind.data.frame(itemnumber,missingalpha)
print("The reliability of the test if each item was eliminated is: ")
```

```
## [1] "The reliability of the test if each item was eliminated is: "
```

```
print(partialalphas)
```

```
##    itemnumber missingalpha
## 1           1       0.7262
## 2           2       0.7315
## 3           3       0.7222
## 4           4       0.7256
## 5           5       0.7316
## 6           6       0.7317
## 7           7       0.7334
## 8           8       0.7251
## 9           9       0.7249
## 10         10       0.7189
## 11         11       0.7313
## 12         12       0.7307
## 13         13       0.7354
## 14         14       0.7313
## 15         15       0.7384
## 16         16       0.7325
## 17         17       0.7308
## 18         18       0.7226
## 19         19       0.7308
## 20         20       0.7253
## 21         21       0.7326
## 22         22       0.7352
## 23         23       0.7376
## 24         24       0.7376
## 25         25       0.7364
## 26         26       0.7368
## 27         27       0.7308
## 28         28       0.7359
## 29         29       0.7356
```

```
#time stamp
print(Sys.time())
```

```
## [1] "2019-10-30 11:48:24 EDT"
```