MK Honors Thesis Dataset Generation

1.0 Setup

In order to synthesize an artificial dataset, we must first load our seed data into R.

```
require(readxl)
```

```
## Loading required package: readxl
```

```
library(readxl)
library(descr)
```

```
## Warning: package 'descr' was built under R version 3.6.3
```

```
Intake <- read_excel("Intake+Questionnaire+for+MK+Honors+Thesis_December+3,+2019_12.02.xlsx")
Daily <- read_excel("Daily+Questionnaire+for+MK+Honors+Thesis_December+3,+2019_11.54.xlsx")
```

The data are read in as character vectors by default rather than categorical factors. We must convert the data typing of each variable to factor before proceeding.

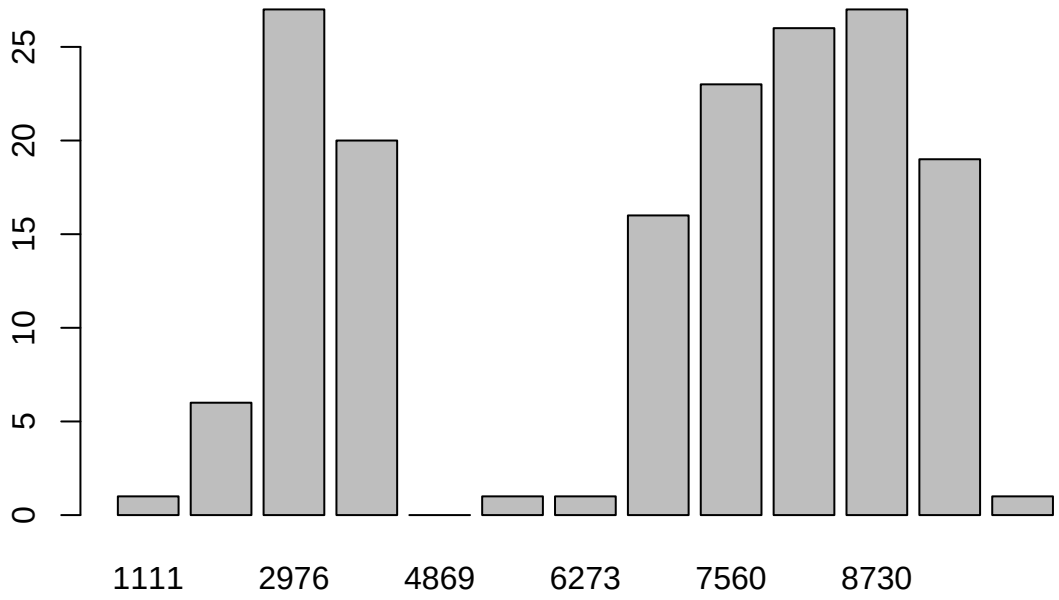
```
Intake2 <- as.data.frame(unclass(Intake))
Daily2 <- as.data.frame(unclass(Daily))
```

Finally, labels and metadata should be stripped from the dataframe. For convenience, we also relabel the Intake2 and Daily2 datasets to `i` and `d` respectively.

```
i <- Intake2[-1,] # renaming and parsing factor labels
d <- Daily2[-1,] # renaming and parsing factor labels

Daily2$Q15[Daily2$Q15 == 4869] <- 4689

freqlist <- freq(Daily2$Q15)
```

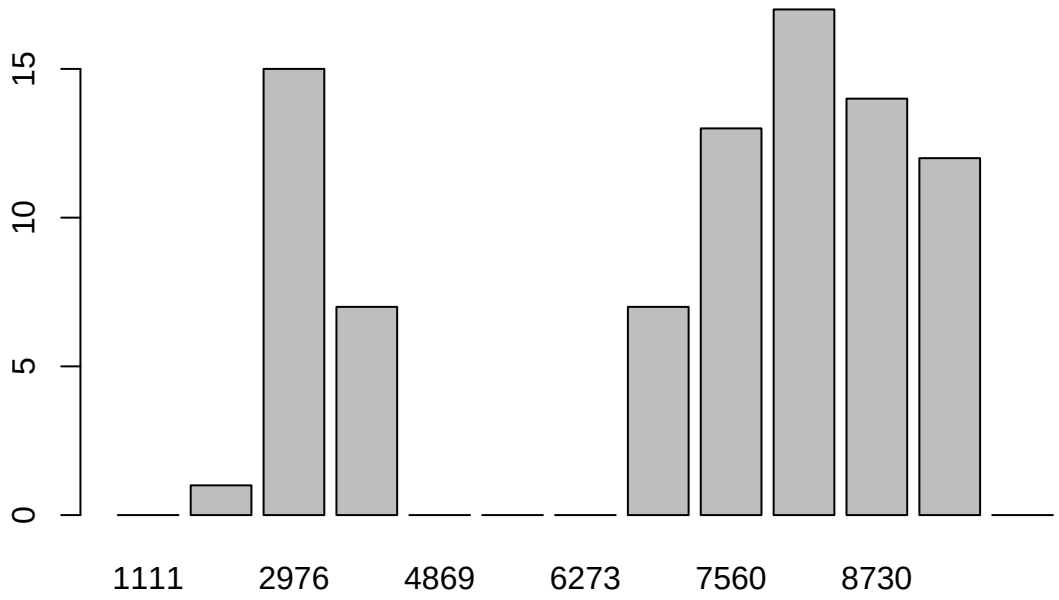



```
f <- which(freqlist[,1] > 1)
f <- names(f)[1:(length(f)-2)]

d <- Daily2[as.character(Daily2$Q15) %in% f,]
d <- d[d$Finished == "True",]

d_drop <- d[!is.na(d$Q17),] # removing NA values


freqlist <- freq(d_drop$Q15)
```




```
f <- which(freqlist[,1] > 1)
f <- names(f)[1:(length(f)-1)]
d_drop <- d_drop[as.character(d_drop$Q15) %in% f,]
d_main <- d_drop[,19:79] # parse metadata
d_main <- na.omit(d_main) ## THIS LINE REMOVES ALL INCOMPLETE CASES ##
d_label <- d_main[,61] # label vector


l_counts <- matrix(0,ncol=5,nrow=5)
for (i in f){
  d_sub <- d_drop[d_drop$Q15 == i,]
  for (row in 2:nrow(d_sub)){
    pastL <- d_sub$Q17[row-1]
    L <- d_sub$Q17[row]
    l_counts[pastL,L] <- l_counts[pastL,L] + 1
    #print(paste(i,row,pastL,L,nrow(d_sub)))
    }
}
roulette_l <- prop.table(l_counts,1)
```

1.1 Artificial Dataset Synthesis

The artificial dataset will be generated using joint distribution roulette wheel sampling.

Variables on daily: 19:78

MOOD: 1 - 20 DAILY FXN: 21 - 30 ACTIVITY FXN: 31 - 39 RELIABILITY FXN: 40 - 44 DIET/HYD FXN: 45 - 47 EXERCISE FXN: 48 - 50 MINDFULNESS/BIOFEED FXN: 51 - 56 SLEEP FXN: 57 - 60 PAIN LVL: 61

```
sampleLabel <- function(day,pastL,d_label,roulette_l){
  if (day == 1){
    rand <- sample(1:length(d_label),1) # randomly sample a label from labels
    L <- d_label[rand] # sample that entry of d_label
    return(L)
  }
  else{
    probs <- roulette_l[pastL,]
    rand <- runif(1)
    for (L in 1:length(probs)){
      if (probs[L] >= rand){
        return(L)
      }
      else{
        rand <- rand - probs[L]
      }
    }
    return(length(probs))
  }
}

createDay <- function(L, d_main, scales, nscales) {
  inst <- c(L) # create new artificial instance with label L
  d_subset <- d_main[d_main$Q17 == L,] # only select cases with label
  for (s_i in 1:nscales){ # instance generation
    start = scales[2*s_i-1]
    end = scales[2*s_i]
    rand <- sample(1:nrow(d_subset),1)
    for(att in start:end){
      inst <- c(inst,as.integer(d_subset[rand,att]) - 2)
    }
  }
  return(inst)
}

rouletteWheelSampling <- function(n)
{
  #Variables
  art <- data.frame()
  scales <- c(c(1,20),c(21,30),c(31,39),c(40,44),c(45,47),c(48,50),c(51,56),c(57,60))
  nscales <- length(scales)/2

  paste('Now Generating',n,'samples',sep=" ")

  for (person in 1:n) {
    L <- 1 # arbitrary
    for (day in 1:28) {
```

```
      L <- sampleLabel(day,L,d_label,roulette_l)
      meas <- createDay(L, d_main, scales, nscales)
      inst <- c(person, day, meas)
      art <- rbind(art, inst)
    }
  }
  my_names <- names(Daily)

  for (i in 1:ncol(Daily)) {
    my_names[i] <- paste0(my_names[i], ": ", Daily[1, i])
  }
  my_names <- my_names[19:79]

  names(art) <- c("participant","day","label",my_names[1:60])
  return(art)
}

art <- rouletteWheelSampling(100)
write.csv(art,'HonorsData.csv',row.names = FALSE)
```