Module 3: Laying the Groundwork for Set-Up of Dependent Variables through Data Visualization

April 24, 2021

Contents

MODULE 3 GOAL: By the end of this module, you will be able to:

- Use the code in `module-03.R` to create a visual snapshot of the number of EMAs and time between EMAs within the data file `dat_smoking` – the data file which we will be using to construct our Dependent Variable.

1 Live Demo: Working with `module-03.R`

1.1 Step 1

```
# How to we get from merged.csv to smoking.csv?
dat_smoking <- dat_big_merged %>%
  select(id:smoking_delta_minutes) %>%
  # We consider the "last assessment" to refer to either of the two situations below:
  # 1. participant-initiated EMAs (any type) having with_any_response=0
  # or with_any_response=1
  # 2. Random EMA having with_any_response=1
  # In other words, the only situation not included in the "last assessment"
  # are those Random EMAs which the participant did not provide any response
  # All rows in merged.csv having
  filter(!is.na(ema_order)) %>%
  # Remember: order according to increasing participant ID
  # and within each participant ID, according to increasing time
  arrange(id, time_unixts)
```

1.2 Step 2

```
# Parameters that may be adjusted

# e.g., if this is 2, then we are selecting the 2nd participant ID in the list
choose_idx <- 38
```

```r
# Minimum number of days since 12AM of start of study date
xlim_min <- 0
# Maximum number of days since 12AM of start of study date
xlim_max <- 28
```

1.3 Step 3

```r
# Do not adjust plotting parameters below this line

if(xlim_min > xlim_max){
  print("Error: xlim_min must be less than or equal to xlim_max")
}

# What are the unique participant IDs which are present in dat_smoking?
participant_ids <- unique(dat_smoking$id)

# Let's visualize the data for one particular participant
current_participant <- participant_ids[choose_idx]

# Take rows corresponding to this particular participant
plotdat_participant <- dat_smoking %>% filter(id == current_participant)

# Create a new time variable (just like we did before)
# that captures the number of days elapsed since 12AM of start of study
plotdat_participant <- plotdat_participant %>%
  mutate(num_secs_elapsed_since_start_study = time_unixts - start_study_unixts) %>%
  mutate(num_hrs_elapsed_since_start_study = num_secs_elapsed_since_start_study/(60*60)) %>%
  mutate(num_days_elapsed_since_start_study = num_hrs_elapsed_since_start_study/24)  %>%
  mutate(roundeddown_num_days_elapsed_since_start_study = floor(num_days_elapsed_since_start_study))

# Layer on each component of the plot
plot(-1, xaxt = "n", yaxt = "n",
     xlab = "Day Since Start of Study ('0' represents midnight on the date when study began)",
     ylab = "",
     xlim = c(xlim_min, xlim_max), ylim = c(0,0.3),
     cex.lab = 2,
     frame.plot = FALSE)

if(xlim_max - xlim_min <=7){
  # Use half-day increments
  axis(1, at = seq(xlim_min, xlim_max + 1 , 0.5), cex.axis = 2, lwd.ticks = 2, gap.axis = 1.2)
}else{
  # Use increments of 7 days
  axis(1, at = seq(xlim_min, xlim_max + 1 , 7), cex.axis = 2, lwd.ticks = 2, gap.axis = 1.2)
}

# Identify which rows correspond to each kind of EMA
plotdat_random <- plotdat_participant %>%
  filter(assessment_type == "Post-Quit Random")

plotdat_urge <- plotdat_participant %>%
  filter(assessment_type == "Post-Quit Urge")
```

```r
plotdat_already_slipped <- plotdat_participant  %>%
  filter(assessment_type == "Post-Quit Already Slipped")

plotdat_part_one <- plotdat_participant %>%
  filter(assessment_type == "Post-Quit About to Slip Part One")

plotdat_part_two <- plotdat_participant %>%
  filter(assessment_type == "Post-Quit About to Slip Part Two")

abline(v = plotdat_random$num_days_elapsed_since_start_study, lty = 2, lwd = 2, col = "red")

# Note that if the number of rows in the plot data is equal to zero,
# then no new points will be added to the existing plot; no error message will be displayed
points(plotdat_random$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_random)),
       pch = 17, cex = 2, col = "black")

points(plotdat_urge$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_urge)),
       pch = 19, cex = 2, col = "orange")

points(plotdat_already_slipped$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_already_slipped)),
       pch = 19, cex = 2, col = "seagreen")

points(plotdat_part_one$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_part_one)),
       pch = 19, cex = 2, col = "lightblue")

points(plotdat_part_two$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_part_two)),
       pch = 19, cex = 2, col = "blue")

# Identify which rows correspond to each kind of EMA
plotdat_random <- plotdat_participant %>%
  filter(assessment_type == "Pre-Quit Random")

plotdat_urge <- plotdat_participant %>%
  filter(assessment_type == "Pre-Quit Urge")

plotdat_part_one <- plotdat_participant %>%
  filter(assessment_type == "Pre-Quit About to Slip Part One")

plotdat_part_two <- plotdat_participant %>%
  filter(assessment_type == "Pre-Quit About to Slip Part Two")

abline(v = plotdat_random$num_days_elapsed_since_start_study, lty = 2, lwd = 2, col = "red")

# Note that if the number of rows in the plot data is equal to zero,
# then no new points will be added to the existing plot; no error message will be displayed
points(plotdat_random$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_random)),
       pch = 17, cex = 2, col = "black")
```

```r
points(plotdat_urge$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_urge)),
       pch = 19, cex = 2, col = "orange")

points(plotdat_already_slipped$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_already_slipped)),
       pch = 19, cex = 2, col = "seagreen")

points(plotdat_part_one$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_part_one)),
       pch = 19, cex = 2, col = "lightblue")

points(plotdat_part_two$num_days_elapsed_since_start_study,
       rep(0.1, nrow(plotdat_part_two)),
       pch = 19, cex = 2, col = "blue")

legend("topright", c("Random", "Urge", "Already Slipped", "Part One", "Part Two"),
       col = c("black", "orange", "seagreen", "lightblue", "blue"),
       pch = c(17, 19, 19, 19, 19), pt.cex = rep(2,5), cex = 1.2)
```

BREAK: Any questions?