Lifestyle Factors Influencing Well-Being Among College Students in a High-Stress Research Program

Josephine Janas Danica Lyktey Kayla Rogan Carolyn Ko

2025-11-12

Mental ill-health is a rising problem, specifically among university students. Research shows that quality sleep and frequent exercise act as protective factors, while poor sleep quality and inactivity increase risk. However, few studies have objectively measured these variables or examined their relationship to individuals' beliefs and behaviors. This study aims to explore that relationship in a sample of college students at a public education institution in New York enrolled in high-stress research programs. The primary research question this study aims to address is: What lifestyle factors are most associated with well-being? Participants completed a pre-assessment to gauge these variables before participation, and then wore a MUSE-S sleep headband and a Fitbit Charge 6 for 7 days and nights, using a daily survey to self-report their scores on various sleep and exercise measures. Finally, participants completed the post-assessment to assess changes in their beliefs and behaviors throughout participation. It was hypothesized that lifestyle factors with a focus on health intervention may be more associated with well-being than non-lifestyle factors. Quantitative results indicated that median step count differed across subjective health status groups and that higher stepcount is correlated with higher sleepscore. Qualitative results revealed that individuals who engaged in restorative coping strategies reported improved well-being and that motivations for participating in health behaviors varied by altruistic versus self-exploratory intentions. Overall, findings supported prior research and demonstrated that targeted behavioral interventions in sleep and exercise can improve mental health. This approach underscores the importance of personalized, flexible health behavior strategies to promote well-being across diverse populations.

Table of contents

```r
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.1     v stringr   1.5.2
v ggplot2   4.0.0     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.1.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
```

```r
library(psych)
```

```
Attaching package: 'psych'
```

```
The following objects are masked from 'package:ggplot2':

    %+%, alpha
```

```
library(knitr)
library(tibble)
library(dplyr)
library(tidyr)
library(scales)     # for number formatting like comma()
```

```
Attaching package: 'scales'

The following objects are masked from 'package:psych':

    alpha, rescale

The following object is masked from 'package:purrr':

    discard

The following object is masked from 'package:readr':

    col_factor
```

```
library(english)    # to convert numbers to words
```

```
Attaching package: 'english'

The following object is masked from 'package:scales':

    ordinal
```

```
library(stringr)    # for text functions like str_c()
library(ggdist)

#source: Importing Data Once (Hei & McCarty, 2025): https://shanemccarty.github.io/FRIplaybook/
```

1 Import

```r
library(readxl)
# Import Excel file
onesevendata <- read_excel(
    "10.27.25.Day1_7.Clean.xlsx",
    col_names = TRUE)

onesevendata[onesevendata == -99] <- NA
onesevendata[onesevendata == -50] <- NA

##explanation: all -99 and -50 data will be treated as missing data
# View first 10 rows
head(onesevendata, 10)
```

```
# A tibble: 10 x 78
   StartDate           EndDate             Status IPAddress       Progress
   <dttm>              <dttm>               <dbl> <chr>              <dbl>
 1 2025-06-30 15:29:14 2025-06-30 17:10:54      0 64.128.175.42        100
 2 2025-06-30 15:52:41 2025-06-30 18:54:44      0 149.125.91.33        100
 3 2025-06-30 21:07:27 2025-06-30 21:16:25      0 153.33.244.42        100
 4 2025-06-30 23:33:05 2025-06-30 23:38:26      0 149.125.195.32       100
 5 2025-06-24 16:55:07 2025-06-24 16:55:38      0 24.47.129.138         22
 6 2025-07-06 21:21:04 2025-07-07 18:10:49      0 64.128.175.42        100
 7 2025-07-08 07:04:12 2025-07-08 18:09:29      0 149.125.88.193       100
 8 2025-07-09 04:53:03 2025-07-09 05:14:22      0 166.194.188.15       100
 9 2025-08-25 12:24:10 2025-08-25 12:25:47      1 <NA>                 100
10 2025-08-29 10:22:06 2025-08-29 10:25:02      1 <NA>                 100
# i 73 more variables: `Duration (in seconds)` <dbl>, Finished <dbl>,
#   RecordedDate <dttm>, ResponseId <chr>, RecipientLastName <lgl>,
#   RecipientFirstName <lgl>, RecipientEmail <lgl>, ExternalReference <lgl>,
#   LocationLatitude <dbl>, LocationLongitude <dbl>, DistributionChannel <chr>,
#   UserLanguage <chr>, Q_RecaptchaScore <dbl>, SURVEYDAY <dbl>,
#   PASSWORD_COLOR <dbl>, PASSWORD <chr>, `7DAYS` <dbl>, YEAR <dbl>,
#   PROGRAM <dbl>, LIVING <dbl>, `GENDER ` <dbl>, SEXUALIDENTITY <dbl>, ...
```

```r
#source: Importing Data Once (Hei & McCarty, 2025): https://shanemccarty.github.io/FRIplaybook/
```

```r
library(tidyr)

## Convert to wide format
wide_onesevendata <- onesevendata %>%
  pivot_wider(
    id_cols = PASSWORD,
    names_from = SURVEYDAY,
    values_from = c(`MENTALHEALTHSTATUS`, `HEALTHSTATUS`),
    names_glue = "{.value}_T{SURVEYDAY}"
  )
```

```
#source: https://dcl-prog.stanford.edu/list-columns.html
#source: Tidying your Data (McCarty et. al., 2025): https://shanemccarty.github.io/FRIplaybook/
#explanation: this allows data to be viewed with only one row per participant, allowin for with
```

```
library(readxl)
# Import Excel file
daily_survey_clean <- read_excel(
    "daily_survey_clean.xlsx",
    col_names = TRUE)

#source: Importing Data Once (Hei & McCarty, 2025): https://shanemccarty.github.io/FRIplaybook/
#explanation: all -99 and -50 data will be treated as missing data
```

1.1 Combining Daily Survey with Mental Health Status and Health Status

```
library(readxl)
library(dplyr)


# Select only the variables you need from secondary dataset, then join
masterdata <- wide_onesevendata %>%
  left_join(
    daily_survey_clean %>% select("PASSWORD", "SLEEPSCORE_T1", "SLEEPSCORE_T2", "SLEEPSCORE_T3"
    by = "PASSWORD"
  )

#source: Tidying your Data (McCarty et. al., 2025): https://shanemccarty.github.io/FRIplaybook/
#explanation: join data from the day 1/7 survey to data fom the daily survey
```

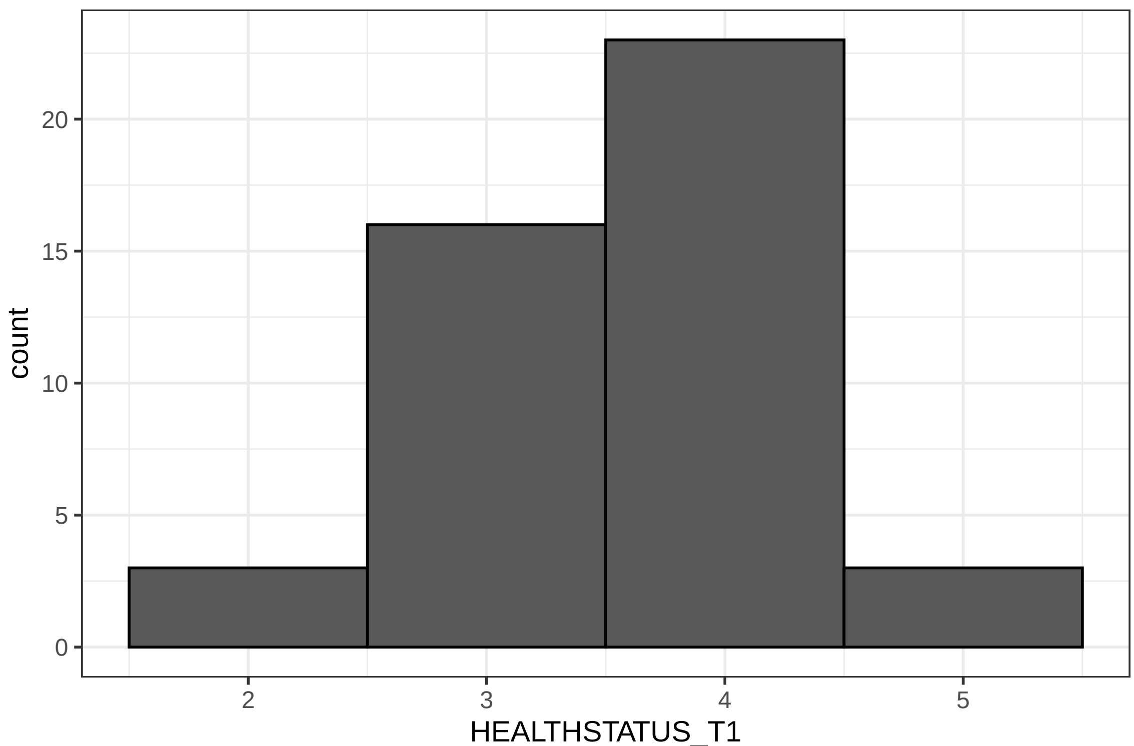
```
library(dplyr)
library(ggplot2)

# Fix list columns in wide_onesevendata
library(dplyr)

wide_onesevendata <- wide_onesevendata %>%
  mutate(across(starts_with("MENTALHEALTHSTATUS_"),
                ~ as.numeric(as.character(sapply(., `[`, 1)))))

wide_onesevendata <- wide_onesevendata %>%
  mutate(across(starts_with("HEALTHSTATUS_"),
                ~ as.numeric(as.character(sapply(., `[`, 1)))))

#source: https://r4ds.hadley.nz/data-transform.html
#explanation: this makes it so that the list is viewed as numbers instead of characters
```

```r
# Convert HEALTHSTATUS_T1 from list to numeric
library(dplyr)
masterdata <- masterdata %>%
  mutate(HEALTHSTATUS_T1 = as.numeric(as.character(sapply(HEALTHSTATUS_T1, `[`, 1))))
```

```
Warning: There was 1 warning in `mutate()`.
i In argument: `HEALTHSTATUS_T1 =
  as.numeric(as.character(sapply(HEALTHSTATUS_T1, `[`, 1)))`.
Caused by warning:
! NAs introduced by coercion
```

```r
#check health status distribution
ggplot(masterdata, aes(x = HEALTHSTATUS_T1)) +
  geom_histogram(binwidth = 1, color = "black") +
  theme_bw()
```

```
Warning: Removed 7 rows containing non-finite outside the scale range
(`stat_bin()`).
```



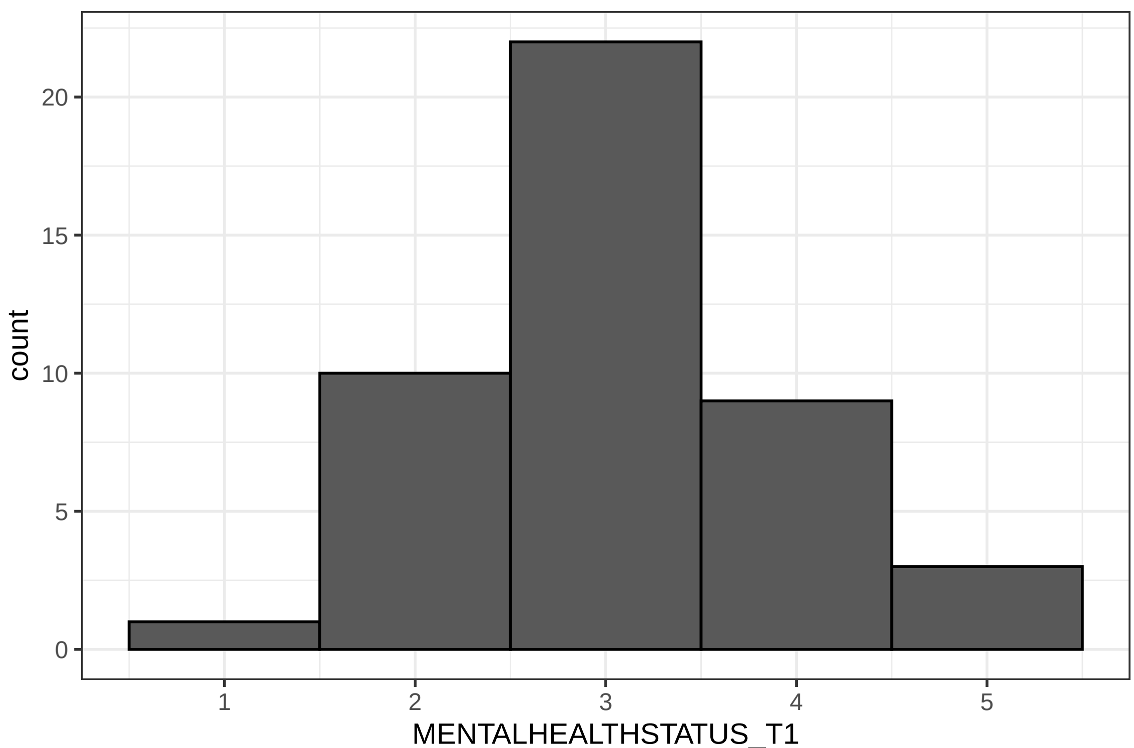

```r
#source: https://r4ds.hadley.nz/data-transform.html
#explanation: extract first element from list and convert to numeric
```

```
# Convert MENTALHEALTHSTATUS_T1 from list to numeric
library(dplyr)
masterdata <- masterdata %>%
  mutate(MENTALHEALTHSTATUS_T1 = as.numeric(as.character(sapply(MENTALHEALTHSTATUS_T1, `[`, 1)
```

```
Warning: There was 1 warning in `mutate()`.
i In argument: `MENTALHEALTHSTATUS_T1 =
  as.numeric(as.character(sapply(MENTALHEALTHSTATUS_T1, `[`, 1)))`.
Caused by warning:
! NAs introduced by coercion
```

```
#check mental health status distribution
ggplot(masterdata, aes(x = MENTALHEALTHSTATUS_T1)) +
  geom_histogram(binwidth = 1, color = "black") +
  theme_bw()
```

```
Warning: Removed 7 rows containing non-finite outside the scale range
(`stat_bin()`).
```




```
#source: https://r4ds.hadley.nz/data-transform.html
#explanation: extract first element from list and convert to numeric
```

```
recode_labels <- function(x) {
  case_when(
    x == 1 ~ "Poor",
    x == 2 ~ "Fair",
    x == 3 ~ "Good",
    x == 4 ~ "Very Good",
    x == 5 ~ "Excellent",
    TRUE ~ NA_character_
  )
}

#source:
#explanation:
```

1.2 Create Composite Score for Sleep Scores

```
library(psych)
SLEEPSCORE_keys <- list(
  SLEEPSCORE = c("SLEEPSCORE_T2", "SLEEPSCORE_T3", "SLEEPSCORE_T4", "SLEEPSCORE_T5", "SLEEPSCO
)

#source: Creating Composite Scorex from Multi-Item Measures (McCarty, 2025): https://shanemcca
#explanation: create keys for composite scoring, a list which tells R which survey questions be
#note: remove T1 because all NAs
```

```
library(dplyr)

# Convert all SLEEPSCORE columns to numeric using dplyr
masterdata <- masterdata %>%
  mutate(across(starts_with("SLEEPSCORE_T"), ~ as.numeric(as.character(.))))

# Verify conversion
cat("Checking SLEEPSCORE column types:\n")
```

```
Checking SLEEPSCORE column types:
```

```
masterdata %>%
  select(starts_with("SLEEPSCORE_T")) %>%
  sapply(class) %>%
  print()
```

```
SLEEPSCORE_T1 SLEEPSCORE_T2 SLEEPSCORE_T3 SLEEPSCORE_T4 SLEEPSCORE_T5
    "numeric"     "numeric"     "numeric"     "numeric"     "numeric"
SLEEPSCORE_T6 SLEEPSCORE_T7
    "numeric"     "numeric"
```

```
#source: https://r4ds.hadley.nz/data-transform.html
#explanation: convert character columns to numeric for composite scoring
```

```r
library(psych)

# Now use scoreItems with clean numeric data
SLEEPSCORE_scores <- scoreItems(SLEEPSCORE_keys, masterdata)

# Add composite score to dataframe
masterdata$SLEEPSCORE <- SLEEPSCORE_scores$scores[, "SLEEPSCORE"]

# View reliability statistics
print(SLEEPSCORE_scores)
```

```
Call: scoreItems(keys = SLEEPSCORE_keys, items = masterdata)

(Unstandardized) Alpha:
      SLEEPSCORE
alpha       0.41

Standard errors of unstandardized Alpha:
      SLEEPSCORE
ASE         0.15

Average item correlation:
          SLEEPSCORE
average.r        0.1

Median item correlation:
SLEEPSCORE
     0.073

 Guttman 6* reliability:
         SLEEPSCORE
Lambda.6       0.45

Signal/Noise based upon av.r :
             SLEEPSCORE
Signal/Noise        0.7

Scale intercorrelations corrected for attenuation
 raw correlations below the diagonal, alpha on the diagonal
 corrected correlations above the diagonal:
           SLEEPSCORE
SLEEPSCORE       0.41
```

```
 Average adjusted correlations within and between scales (MIMS)
[1] 0.1

 Average adjusted item x scale correlations within and between scales (MIMT)
[1] 0.5

 In order to see the item by scale loadings and frequency counts of the data
 print with the short option = FALSE
```

```
### IMPORTANT: Please shift from print to ggsave() in FRI playbook so we have consistency acro
### You can do it both ways, but I would prefer to the ggsave way to give us more flexibility.



#source: Creating Composite Scores from Multi-Item Measures (McCarty, 2025): https://shanemcca
#explanation: use scoreItems to calculate composite scores and reliability statistics
```

```
library(psych)
STEPCOUNT_keys <- list(
  STEPCOUNT = c("STEPCOUNT_T1", "STEPCOUNT_T2", "STEPCOUNT_T3", "STEPCOUNT_T4", "STEPCOUNT_T5"
)

#source: Creating Composite Scores from Multi-Item Measures (McCarty, 2025): https://shanemcca
#explanation: create keys for composite scoring, a list which tells R which survey questions be
```

```
library(dplyr)

# Convert all STEPCOUNT columns to numeric using dplyr
masterdata <- masterdata %>%
  mutate(across(starts_with("STEPCOUNT_T"), ~ as.numeric(as.character(.))))

# Verify conversion
cat("Checking STEPCOUNT column types:\n")
```

```
Checking STEPCOUNT column types:
```

```
masterdata %>%
  select(starts_with("STEPCOUNT_T")) %>%
  sapply(class) %>%
  print()
```

```
STEPCOUNT_T1 STEPCOUNT_T2 STEPCOUNT_T3 STEPCOUNT_T4 STEPCOUNT_T5 STEPCOUNT_T6
   "numeric"    "numeric"    "numeric"    "numeric"    "numeric"    "numeric"
STEPCOUNT_T7
   "numeric"
```

```
#source: https://r4ds.hadley.nz/data-transform.html
#explanation: convert character columns to numeric for composite scoring
```

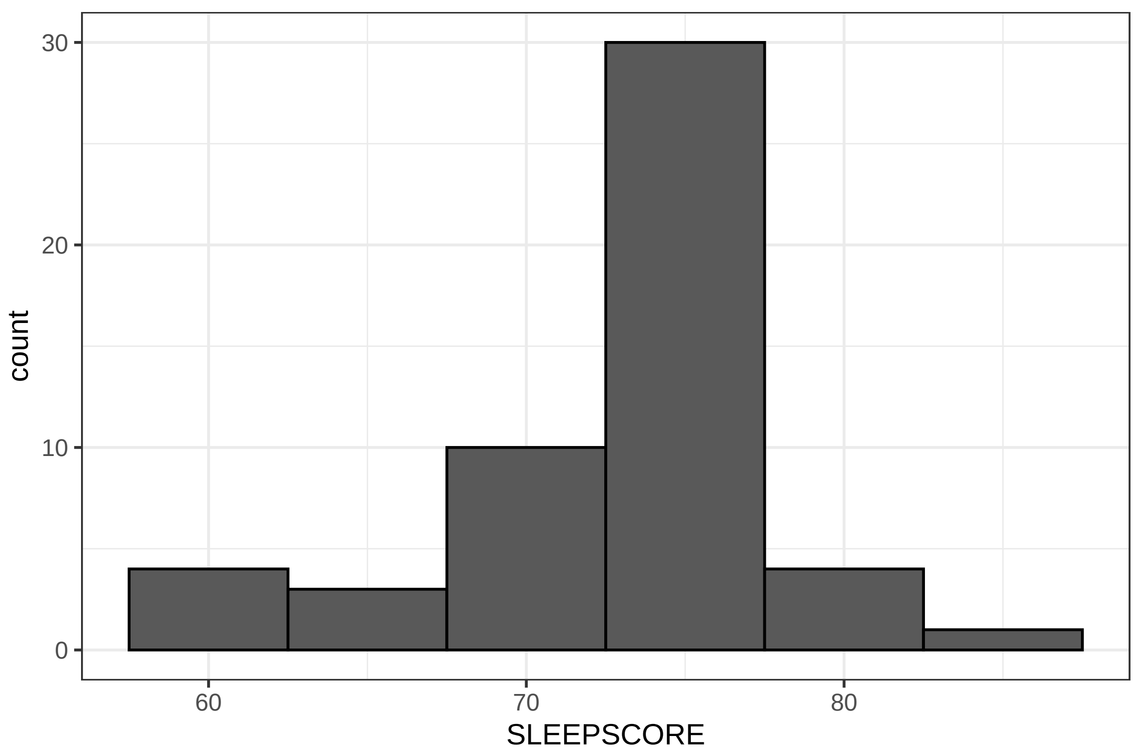
```
library(psych)

# Now use scoreItems with clean numeric data
STEPCOUNT_scores <- scoreItems(STEPCOUNT_keys, masterdata)

# Add composite score to dataframe
masterdata$STEPCOUNT <- STEPCOUNT_scores$scores[, "STEPCOUNT"]

# View reliability statistics
print(STEPCOUNT_scores)
```

```
Call: scoreItems(keys = STEPCOUNT_keys, items = masterdata)

(Unstandardized) Alpha:
      STEPCOUNT
alpha      0.82

Standard errors of unstandardized Alpha:
      STEPCOUNT
ASE       0.066

Average item correlation:
          STEPCOUNT
average.r       0.4

Median item correlation:
STEPCOUNT
     0.43

 Guttman 6* reliability:
         STEPCOUNT
Lambda.6      0.87

Signal/Noise based upon av.r :
             STEPCOUNT
Signal/Noise       4.7

Scale intercorrelations corrected for attenuation
 raw correlations below the diagonal, alpha on the diagonal
 corrected correlations above the diagonal:
          STEPCOUNT
STEPCOUNT      0.82
```

```
 Average adjusted correlations within and between scales (MIMS)
[1] 0.4

 Average adjusted item x scale correlations within and between scales (MIMT)
[1] 0.71

 In order to see the item by scale loadings and frequency counts of the data
 print with the short option = FALSE
```

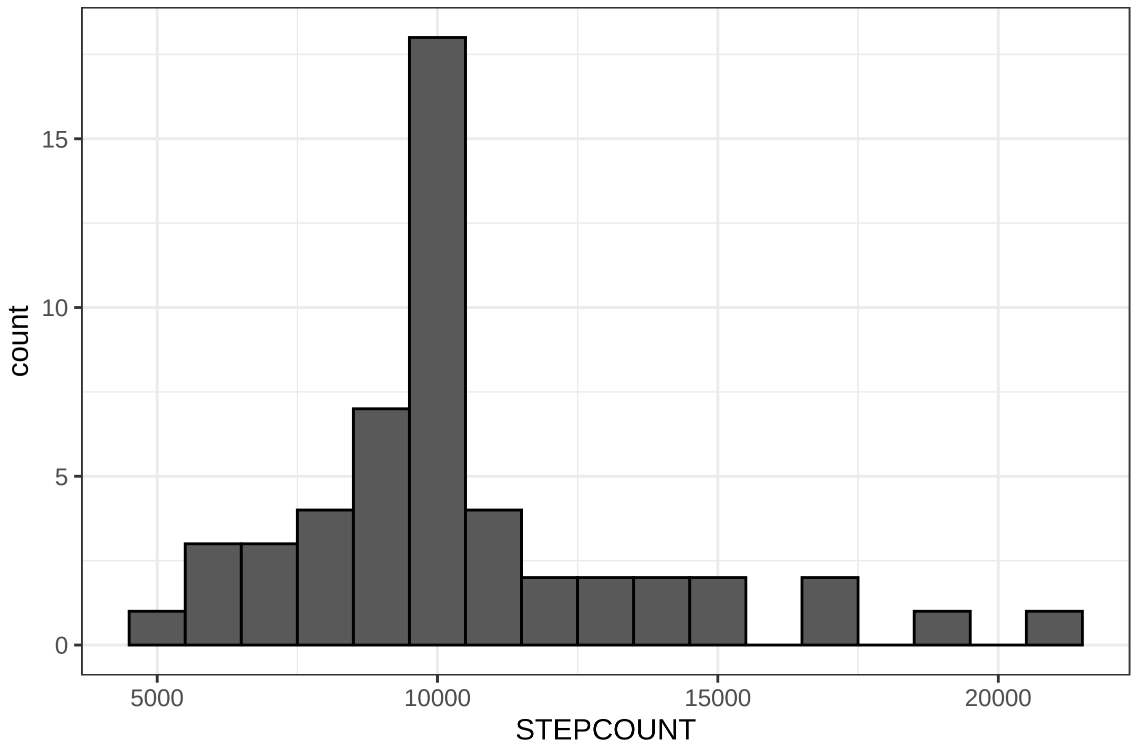
```
#source: Creating Composite Scores from Multi-Item Measures (McCarty, 2025): https://shanemcca
#explanation: use scoreItems to calculate composite scores and reliability statistics
```

```
# Convert SLEEPSCORE from list to numeric
library(dplyr)
masterdata <- masterdata %>%
  mutate(SLEEPSCORE = as.numeric(as.character(sapply(SLEEPSCORE, `[`, 1))))

#check sleep score distribution
ggplot(masterdata, aes(x = SLEEPSCORE)) +
  geom_histogram(binwidth = 5, color = "black") +
  theme_bw()
```




```
#source: https://r4ds.hadley.nz/data-transform.html
#explanation: extract first element from list and convert to numeric
```

```
# Convert STEPCOUNT from list to numeric
library(dplyr)
masterdata <- masterdata %>%
  mutate(STEPCOUNT = as.numeric(as.character(sapply(STEPCOUNT, `[`, 1))))

#check step count distribution
ggplot(masterdata, aes(x = STEPCOUNT)) +
  geom_histogram(binwidth = 1000, color = "black") +
  theme_bw()
```




```
#source: https://r4ds.hadley.nz/data-transform.html
#explanation: extract first element from list and convert to numeric
```

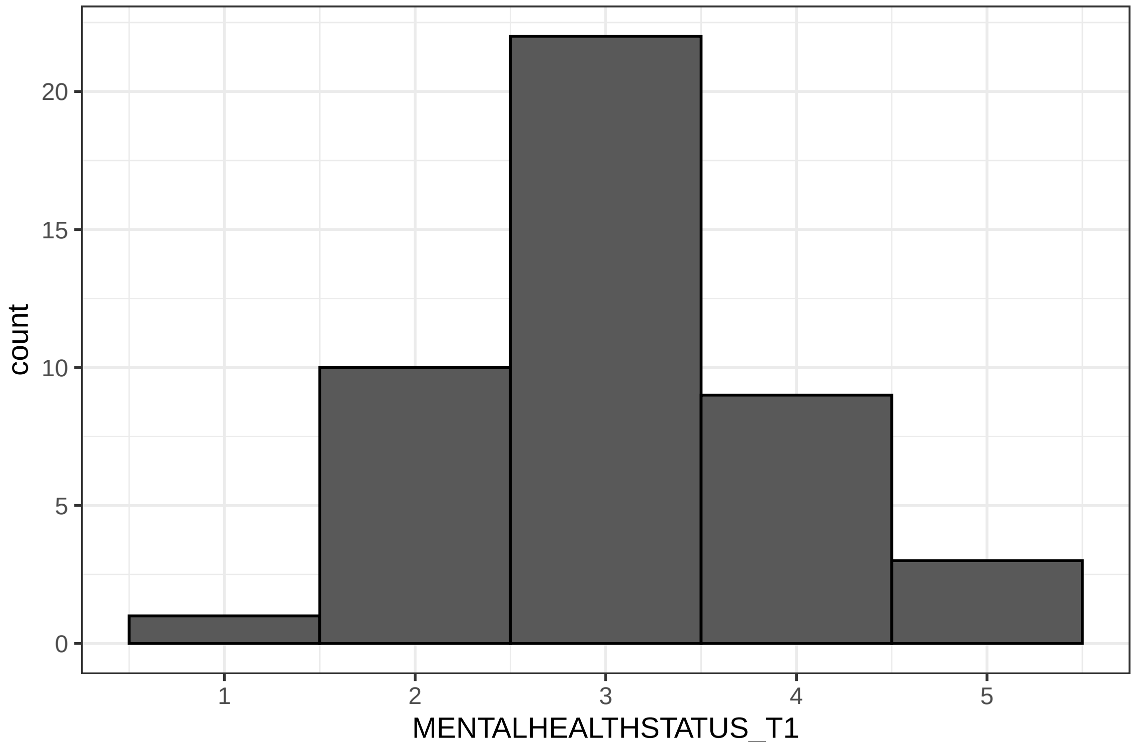
2 Creating Mental Health Status Bar Charts

2.1 Creating Mental Health Status vs. Sleep Score Bar Chart

```
# Convert MENTALHEALTHSTATUS_T1 from list to numeric in masterdata
library(dplyr)
masterdata <- masterdata %>%
  mutate(MENTALHEALTHSTATUS_T1 = as.numeric(as.character(sapply(MENTALHEALTHSTATUS_T1, `[`, 1)

#check mental health status distribution
```

```r
ggplot(masterdata, aes(x = MENTALHEALTHSTATUS_T1)) +
  geom_histogram(binwidth = 1, color = "black") +
  theme_bw()
```

```
Warning: Removed 7 rows containing non-finite outside the scale range
(`stat_bin()`).
```




```r
#source: https://r4ds.hadley.nz/data-transform.html
#explanation: extract first element from list and convert to numeric
```

```r
library(ggplot2)
sleep_mental_health <- ggplot(
  data = masterdata,
  mapping = aes(x = MENTALHEALTHSTATUS_T1, y = SLEEPSCORE)
) +
  ylim(0, 100) +
  geom_bar(stat = 'summary', fun = 'mean', fill = "#673888") +
  ggtitle("Average Sleep Score by Mental Health Status") +
  xlab("Mental Health Status") +
  ylab("MUSE S Sleep Score") +
  theme_bw() +
  scale_x_continuous(
    breaks = 1:5,
    labels = c(
```

```
      "1: Poor",
      "2: Fair",
      "3: Good",
      "4: Very Good",
      "5: Excellent"
    )
  )

#source: datacamp
#explanation:

# Print and save to the plots folder
print(sleep_mental_health)
```

```
Warning: Removed 7 rows containing non-finite outside the scale range
(`stat_summary()`).
```

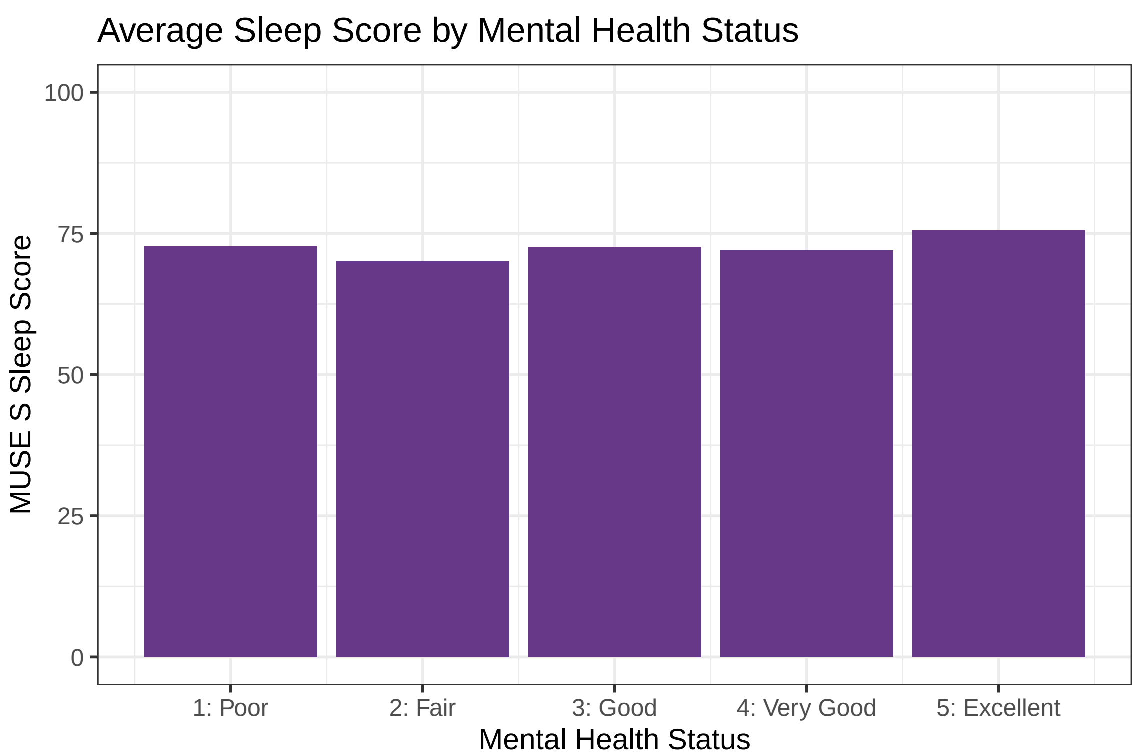


Figure 1: Figure 1. Average Sleep Score by Mental Health Status. The bar graph displays the mean MUSE S sleep scores across five self-reported mental health status categories, ranging from 'Poor' to 'Excellent.' Participants reporting better mental health tended to have slightly higher average sleep scores, though differences across categories appear modest.

```
ggsave("plots/sleep_mental_health.png",
       plot = sleep_mental_health,
       width = 10, height = 8, dpi = 300)
```

```
Warning: Removed 7 rows containing non-finite outside the scale range
(`stat_summary()`).
```

```
#source: Visualize data in ggplot 2 (Silhavy & McCarty, 2025): https://shanemccarty.github.io/
#explanation: save the plot to the plots folder
```

2.2 Creating Mental Health Status vs. Step Count Bar Chart

```
library(ggplot2)
exercise_mental_health <- ggplot(
  data = masterdata,
  mapping = aes(x = MENTALHEALTHSTATUS_T1, y = STEPCOUNT)
) +
  ylim(0, 20000) +
  geom_bar(stat = 'summary', fun = 'mean', fill = "#ef4f91") +
  ggtitle("Average Step Count by Mental Health Status") +
  xlab("Mental Health Status") +
  ylab("FitBit Charge 6 Step Count") +
  theme_bw() +
  scale_x_continuous(
    breaks = 1:5,
    labels = c(
      "1: Poor",
      "2: Fair",
      "3: Good",
      "4: Very Good",
      "5: Excellent"
    )
  )

#source: datacamp
#explanation:

# Print and save to the plots folder
print(exercise_mental_health)
```

```
Warning: Removed 8 rows containing non-finite outside the scale range
(`stat_summary()`).
```

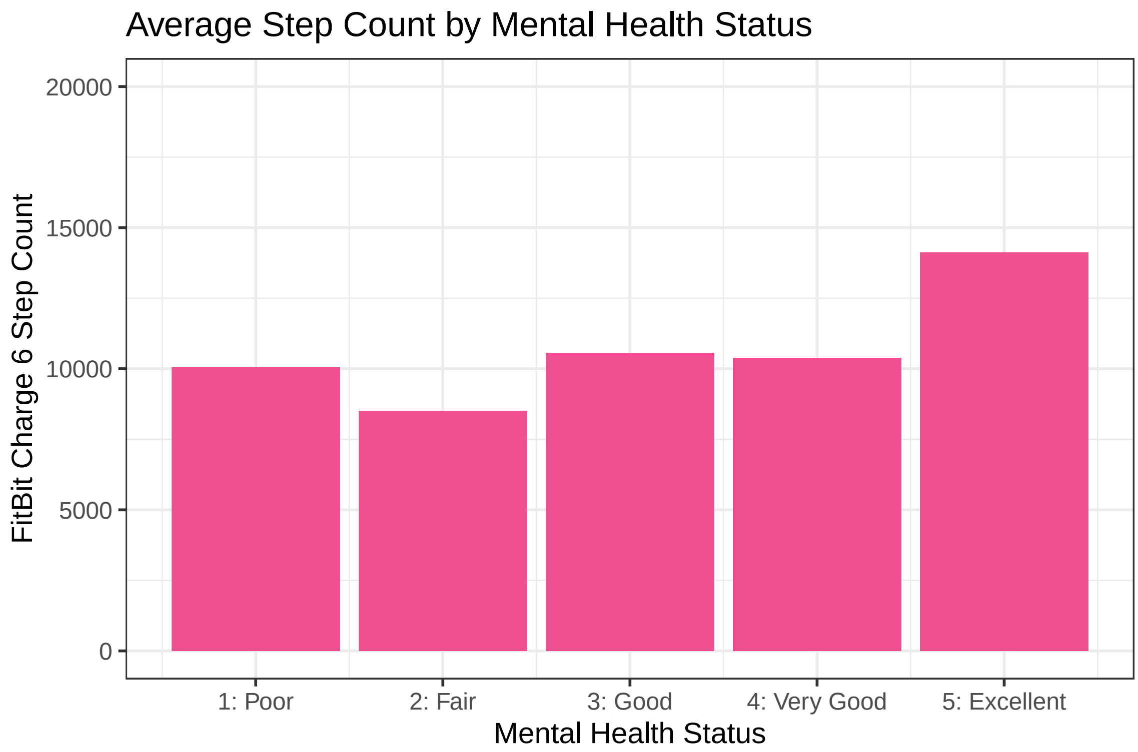


Figure 2: Figure 2. Average Step Count by Mental Health Status. The bar graph illustrates the mean daily step count recorded by the Fitbit Charge 6 across five self-reported mental health categories, from 'Poor' to 'Excellent.' Participants reporting better mental health generally exhibited higher average step counts, with the 'Excellent' group showing the greatest activity levels.

```
ggsave("plots/exercise_mental_health.png",
       plot = exercise_mental_health,
       width = 10, height = 8, dpi = 300)
```

```
Warning: Removed 8 rows containing non-finite outside the scale range
(`stat_summary()`).
```

```
#source: Visualize data in ggplot 2 (Silhavy & McCarty, 2025): https://shanemccarty.github.io/F
#explanation: save the plot to the plots folder
```

3 Creating Health Status Bar Charts

3.1 Creating Health Status v. Sleep Score Bar Chart

```
library(ggplot2)
sleep_health <- ggplot(
```

```
  data = masterdata,
  mapping = aes(x = HEALTHSTATUS_T1, y = SLEEPSCORE)
) +
  ylim(0, 100) +
  geom_bar(stat = 'summary', fun = 'mean', fill = "#673888") +
  ggtitle("Average Sleep Score by Health Status") +
  xlab("Health Status") +
  ylab("MUSE S Sleep Score") +
  theme_bw() +
  scale_x_continuous(
    breaks = 1:5,
    labels = c(
      "1: Poor",
      "2: Fair",
      "3: Good",
      "4: Very Good",
      "5: Excellent"
    )
  )

#source: datacamp
#explanation:

# Print and save to the plots folder
print(sleep_health)
```

```
Warning: Removed 7 rows containing non-finite outside the scale range
(`stat_summary()`).
```

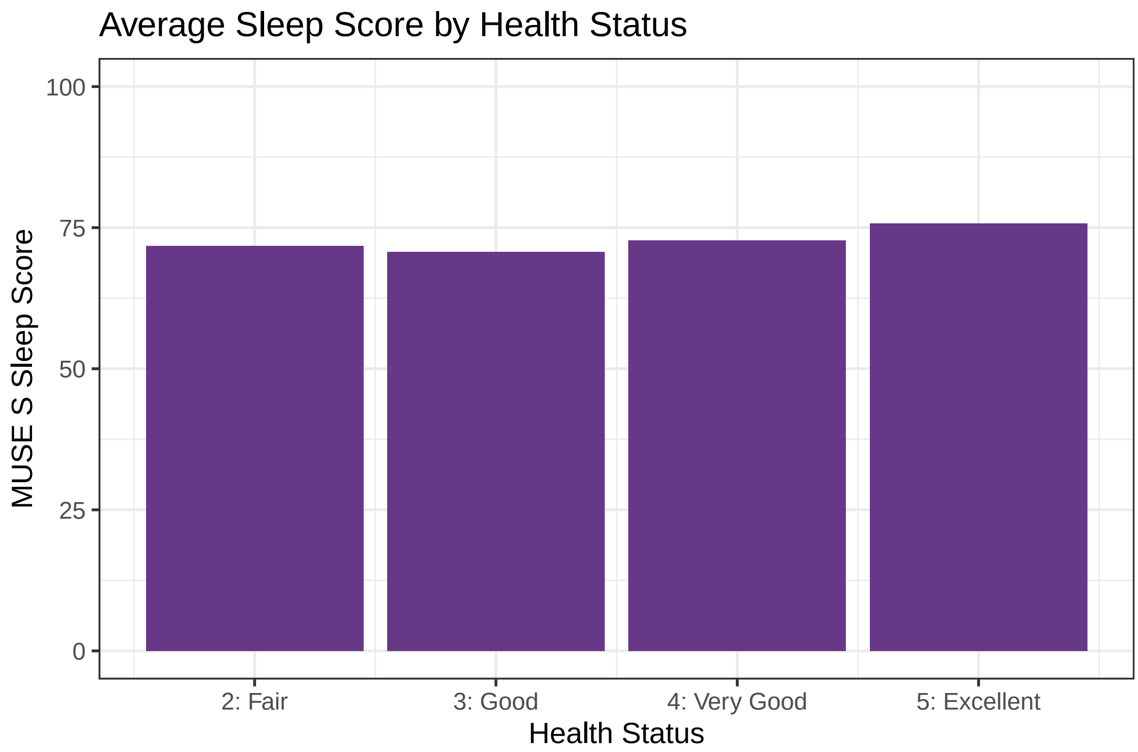


Figure 3: Figure 3. Average Sleep Score by Health Status. The bar graph presents the mean MUSE S sleep scores across four self-reported general health categories: 'Fair,' 'Good,' 'Very Good,' and 'Excellent' ('Poor' was removed because no one selected it). Sleep scores tend to increase slightly as self-rated health improves, indicating a modest positive association between overall health and sleep quality.

```
ggsave("plots/sleep_health.png",
       plot = sleep_health,
       width = 10, height = 8, dpi = 300)
```

```
Warning: Removed 7 rows containing non-finite outside the scale range
(`stat_summary()`).
```

```
#source: Visualize data in ggplot 2 (Silhavy & McCarty, 2025): https://shanemccarty.github.io/F
#explanation: save the plot to the plots folder
```

3.2 Creating Health Status v. Step Count Bar Chart

```
library(dplyr)
library(tidyr)

# Calculate means and 95% confidence intervals for step count by health status
stepcount_health_summary <- masterdata %>%
```

```
  filter(!is.na(HEALTHSTATUS_T1) & !is.na(STEPCOUNT)) %>%
  group_by(HEALTHSTATUS_T1) %>%
  summarise(
    n = n(),
    mean_stepcount = mean(STEPCOUNT, na.rm = TRUE),
    sd_stepcount = sd(STEPCOUNT, na.rm = TRUE),
    se_stepcount = sd_stepcount / sqrt(n),
    ci_lower = mean_stepcount - (1.96 * se_stepcount),
    ci_upper = mean_stepcount + (1.96 * se_stepcount)
  ) %>%
  # Add missing health status level (1: Poor) with NA values
  complete(HEALTHSTATUS_T1 = 1:5,
           fill = list(n = 0, mean_stepcount = NA, sd_stepcount = NA,
                       se_stepcount = NA, ci_lower = NA, ci_upper = NA))

# View the summary statistics
stepcount_health_summary
```

```
# A tibble: 5 x 7
  HEALTHSTATUS_T1     n mean_stepcount sd_stepcount se_stepcount ci_lower
            <dbl> <int>          <dbl>        <dbl>        <dbl>    <dbl>
1               1     0             NA           NA           NA       NA
2               2     3           8914.         987.         570.    7798.
3               3    16          10690.        2762.         691.    9337.
4               4    23           9691.        2882.         601.    8514.
5               5     3          17887.        4199.        2424.   13135.
# i 1 more variable: ci_upper <dbl>
```

```
#source: https://r4ds.hadley.nz/data-transform.html
#explanation: calculate descriptive statistics and 95% confidence intervals for error bars; use
```

```
library(ggplot2)

step_health <- ggplot(
  data = stepcount_health_summary,
  mapping = aes(x = HEALTHSTATUS_T1, y = mean_stepcount)
) +
  ylim(0, 20000) +
  geom_bar(stat = 'identity', fill = "#ef4f91") +
  geom_errorbar(
    aes(ymin = ci_lower, ymax = ci_upper),
    width = 0.2,
    linewidth = 0.7
  ) +
  ggtitle("Average Step Count by Health Status") +
  xlab("Health Status") +
```

```r
  ylab("FitBit Charge 6 Step Count") +
  theme_bw() +
  scale_x_continuous(
    breaks = c(1, 2, 3, 4, 5),
    labels = c(
      "1: Poor",
      "2: Fair",
      "3: Good",
      "4: Very Good",
      "5: Excellent"
    )
  )

#source: https://ggplot2.tidyverse.org/reference/geom_errorbar.html
#explanation: create bar chart with 95% confidence interval error bars showing variability in s

step_health
```

```
Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_bar()`).
```

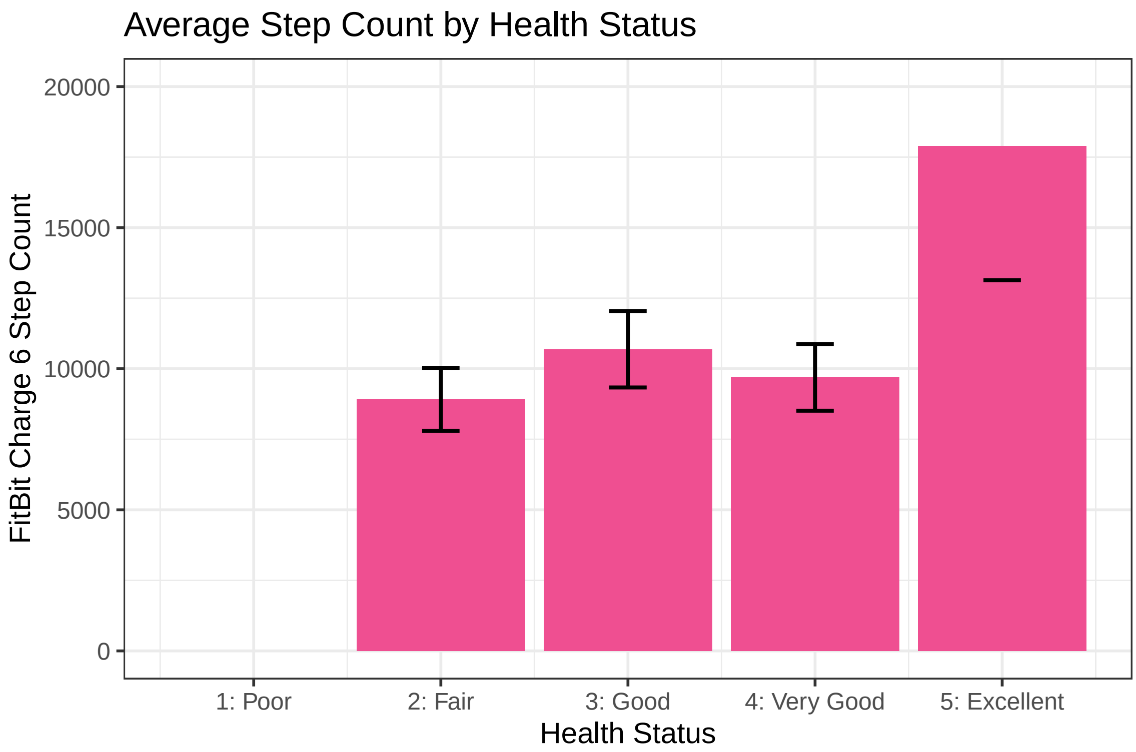


Figure 4: Figure 4. Average Step Count by Health Status. The bar graph shows the mean Fitbit Charge 6 step count across five self-reported general health categories from 'Poor' to 'Excellent.' No participants selected 'Poor,' so this category appears empty. Error bars represent 95% confidence intervals. The 'Excellent' group shows notably higher step counts, though this group has only 3 participants and wide confidence intervals.

```
ggsave("plots/step_health.png",
       plot = step_health,
       width = 10, height = 8, dpi = 300)
```

```
Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_bar()`).
```

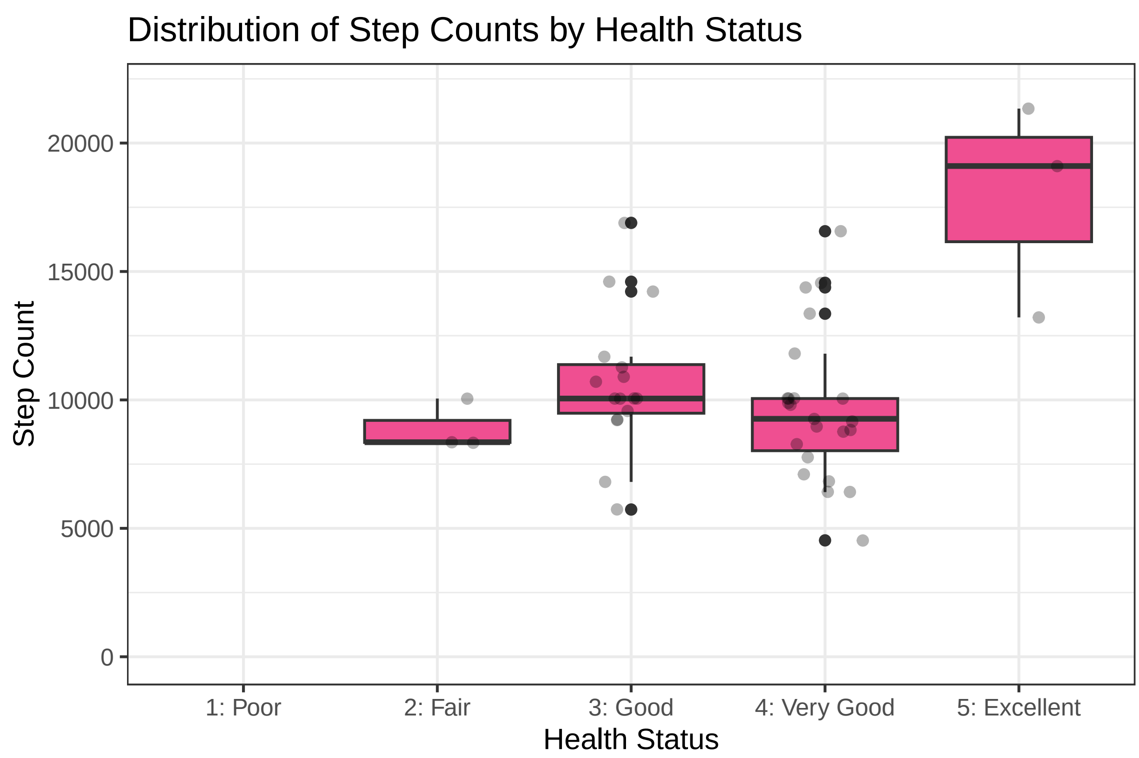
```
#source: Visualize data in ggplot 2 (Silhavy & McCarty, 2025): https://shanemccarty.github.io/I
#explanation: save the plot to the plots folder
```

4 Creating Health Status Box Plots

4.1 Creating Health Status v. Step Count Box Plot

```
library(dplyr)
library(tidyr)
```

```r
library(ggplot2)

# Check sample sizes and data distribution
samples.stepsXhealth <- masterdata %>%
  filter(!is.na(HEALTHSTATUS_T1) & !is.na(STEPCOUNT)) %>%
  group_by(HEALTHSTATUS_T1) %>%
  summarise(
    n = n(),
    mean = mean(STEPCOUNT),
    median = median(STEPCOUNT),
    sd = sd(STEPCOUNT),
    min = min(STEPCOUNT),
    max = max(STEPCOUNT)
  ) %>%
  # Add missing health status level (1: Poor)
  complete(HEALTHSTATUS_T1 = 1:5,
           fill = list(n = 0, mean = NA, median = NA, sd = NA, min = NA, max = NA))

## to check statistics used in the boxplot, remove # below
#samples.stepsXhealth

# Prepare data with all health status levels
plot_data <- masterdata %>%
  filter(!is.na(HEALTHSTATUS_T1) & !is.na(STEPCOUNT)) %>%
  mutate(HEALTHSTATUS_T1 = factor(HEALTHSTATUS_T1, levels = 1:5))

# Visualize the distribution with a boxplot to see outliers
plot.stepsXhealth <- ggplot(plot_data, aes(x = HEALTHSTATUS_T1, y = STEPCOUNT)) +
  geom_boxplot(fill = "#ef4f91") +
  geom_jitter(width = 0.2, alpha = 0.3) +
  ggtitle("Distribution of Step Counts by Health Status") +
  xlab("Health Status") +
  ylab("Step Count") +
  theme_bw() +
  scale_x_discrete(
    labels = c(
      "1: Poor",
      "2: Fair",
      "3: Good",
      "4: Very Good",
      "5: Excellent"
    ),
    drop = FALSE
  ) +
  ylim(0, 22000)

plot.stepsXhealth
```

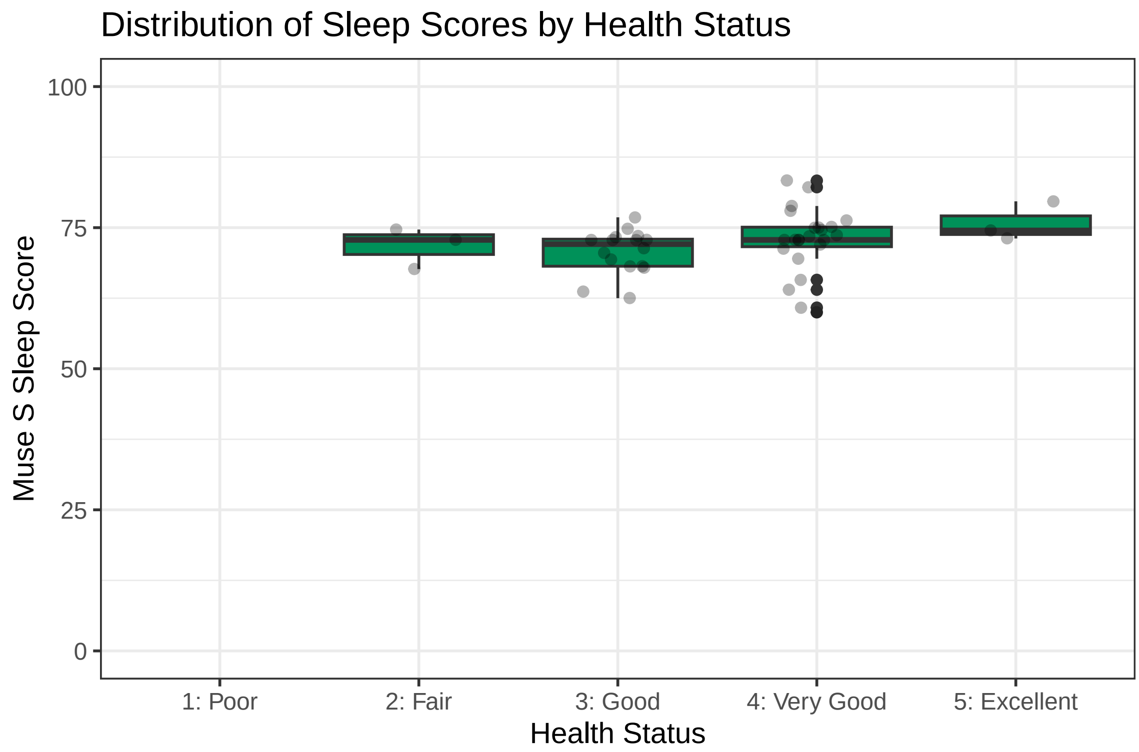



```
#source: https://ggplot2.tidyverse.org/reference/scale_discrete.html
#explanation: boxplot showing distribution of step counts with all health status categories in
```

4.2 Creating Health Status v. Sleep Score Box Plot

```
library(dplyr)
library(tidyr)
library(ggplot2)

# Check sample sizes and data distribution
samples.sleepXhealth <- masterdata %>%
  filter(!is.na(HEALTHSTATUS_T1) & !is.na(SLEEPSCORE)) %>%
  group_by(HEALTHSTATUS_T1) %>%
  summarise(
    n = n(),
    mean = mean(SLEEPSCORE),
    median = median(SLEEPSCORE),
    sd = sd(SLEEPSCORE),
    min = min(SLEEPSCORE),
    max = max(SLEEPSCORE)
  ) %>%
  # Add missing health status level (1: Poor)
  complete(HEALTHSTATUS_T1 = 1:5,
           fill = list(n = 0, mean = NA, median = NA, sd = NA, min = NA, max = NA))
```

```r
## to check statistics used in the boxplot, remove # below
#samples.stepsXhealth

# Prepare data with all health status levels
plot_data <- masterdata %>%
  filter(!is.na(HEALTHSTATUS_T1) & !is.na(SLEEPSCORE)) %>%
  mutate(HEALTHSTATUS_T1 = factor(HEALTHSTATUS_T1, levels = 1:5))

# Visualize the distribution with a boxplot to see outliers
plot.sleepXhealth <- ggplot(plot_data, aes(x = HEALTHSTATUS_T1, y = SLEEPSCORE)) +
  geom_boxplot(fill = "#009159") +
  geom_jitter(width = 0.2, alpha = 0.3) +
  ggtitle("Distribution of Sleep Scores by Health Status") +
  xlab("Health Status") +
  ylab("Muse S Sleep Score") +
  theme_bw() +
  scale_x_discrete(
    labels = c(
      "1: Poor",
      "2: Fair",
      "3: Good",
      "4: Very Good",
      "5: Excellent"
    ),
    drop = FALSE
  ) +
  ylim(0, 100)

plot.sleepXhealth
```



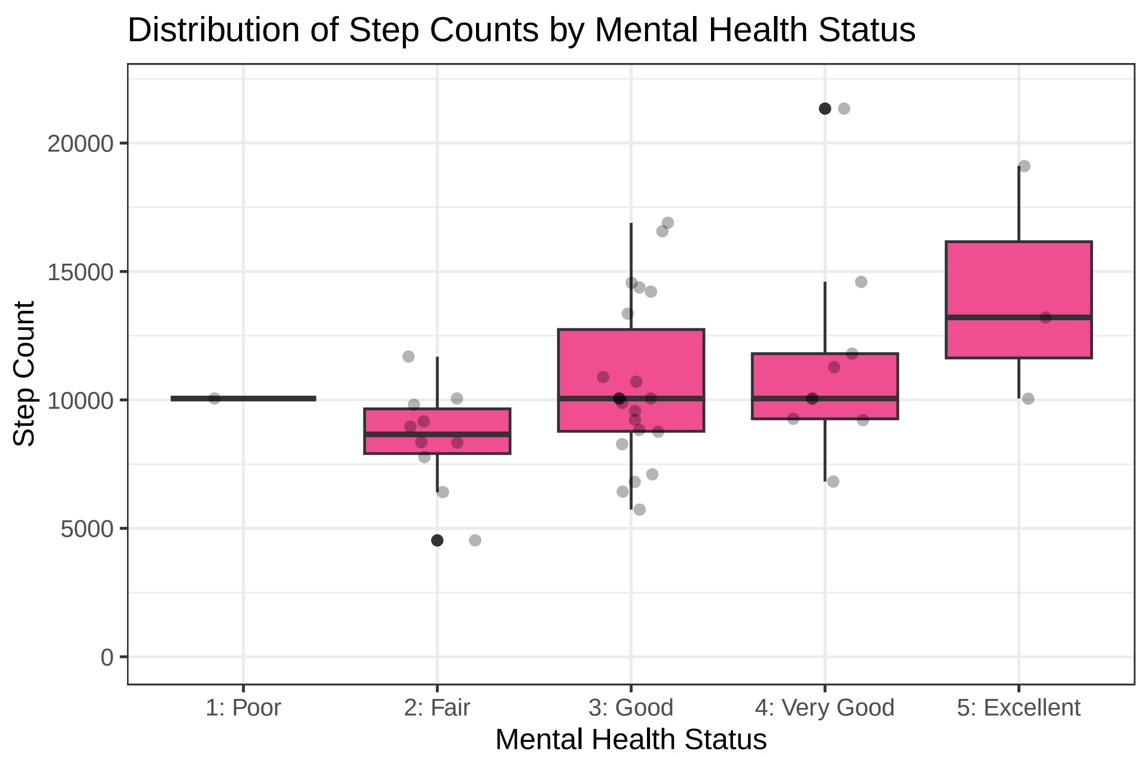

```
#source: https://ggplot2.tidyverse.org/reference/scale_discrete.html
#explanation: boxplot showing distribution of step counts with all health status categories inc
```

5 Creating Mental Health Status Box Plots

5.1 Creating Mental Health Status v. Step Count Box Plot

```
library(dplyr)
library(tidyr)
library(ggplot2)

# Check sample sizes and data distribution
samples.stepsXmentalhealth <- masterdata %>%
  filter(!is.na(MENTALHEALTHSTATUS_T1) & !is.na(STEPCOUNT)) %>%
  group_by(MENTALHEALTHSTATUS_T1) %>%
  summarise(
    n = n(),
    mean = mean(STEPCOUNT),
    median = median(STEPCOUNT),
    sd = sd(STEPCOUNT),
    min = min(STEPCOUNT),
    max = max(STEPCOUNT)
  ) %>%
```

```
  complete(MENTALHEALTHSTATUS_T1 = 1:5,
           fill = list(n = 0, mean = NA, median = NA, sd = NA, min = NA, max = NA))

## to check statistics used in the boxplot, remove # below
#samples.stepsXhealth

# Prepare data with all mental health status levels
plot_data <- masterdata %>%
  filter(!is.na(MENTALHEALTHSTATUS_T1) & !is.na(STEPCOUNT)) %>%
  mutate(MENTALHEALTHSTATUS_T1 = factor(MENTALHEALTHSTATUS_T1, levels = 1:5))

# Visualize the distribution with a boxplot to see outliers
plot.stepsXmentalhealth <- ggplot(plot_data, aes(x = MENTALHEALTHSTATUS_T1, y = STEPCOUNT)) +
  geom_boxplot(fill = "#ef4f91") +
  geom_jitter(width = 0.2, alpha = 0.3) +
  ggtitle("Distribution of Step Counts by Mental Health Status") +
  xlab("Mental Health Status") +
  ylab("Step Count") +
  theme_bw() +
  scale_x_discrete(
    labels = c(
      "1: Poor",
      "2: Fair",
      "3: Good",
      "4: Very Good",
      "5: Excellent"
    ),
    drop = FALSE
  ) +
  ylim(0, 22000)

plot.stepsXmentalhealth
```

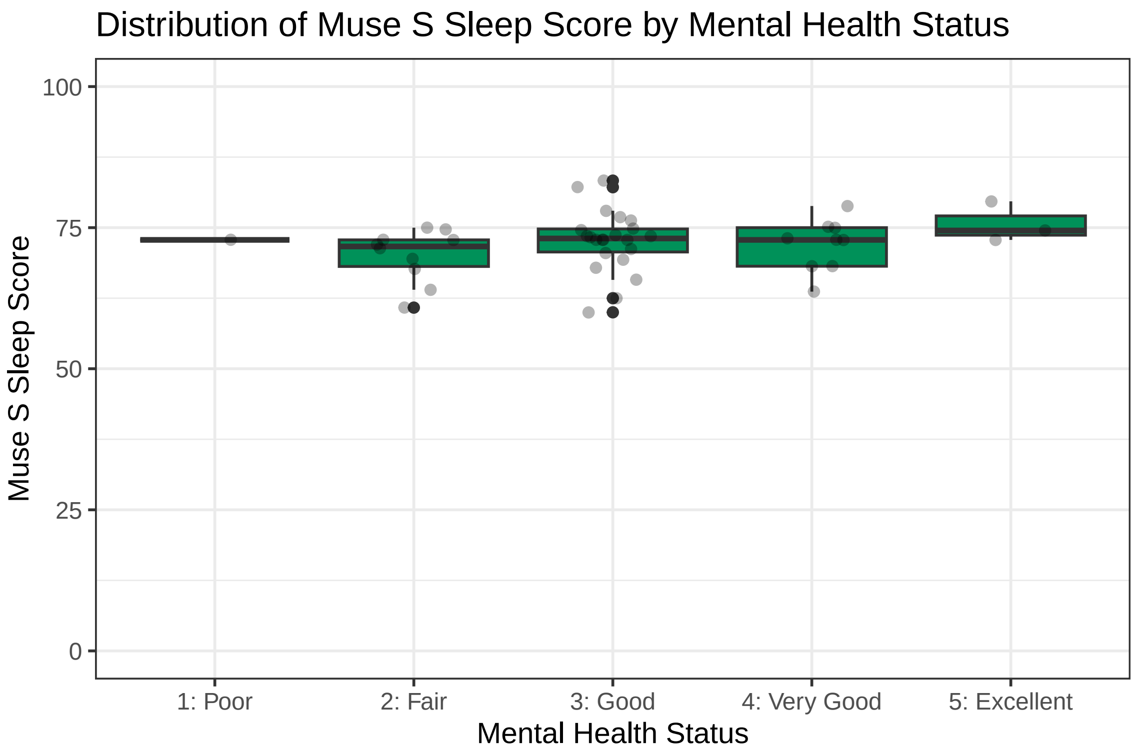



```
#source: https://ggplot2.tidyverse.org/reference/scale_discrete.html
#explanation: boxplot showing distribution of step counts with all health status categories inc
```

5.2 Creating Mental Health Status v. Sleep Score Box Plot

```
library(dplyr)
library(tidyr)
library(ggplot2)

# Check sample sizes and data distribution
samples.sleepXmentalhealth <- masterdata %>%
  filter(!is.na(MENTALHEALTHSTATUS_T1) & !is.na(SLEEPSCORE)) %>%
  group_by(MENTALHEALTHSTATUS_T1) %>%
  summarise(
    n = n(),
    mean = mean(SLEEPSCORE),
    median = median(SLEEPSCORE),
    sd = sd(SLEEPSCORE),
    min = min(SLEEPSCORE),
    max = max(SLEEPSCORE)
  ) %>%

  complete(MENTALHEALTHSTATUS_T1 = 1:5,
           fill = list(n = 0, mean = NA, median = NA, sd = NA, min = NA, max = NA))
```

```r
## to check statistics used in the boxplot, remove # below
#samples.stepsXhealth

# Prepare data with all mental health status levels
plot_data <- masterdata %>%
  filter(!is.na(MENTALHEALTHSTATUS_T1) & !is.na(SLEEPSCORE)) %>%
  mutate(MENTALHEALTHSTATUS_T1 = factor(MENTALHEALTHSTATUS_T1, levels = 1:5))

# Visualize the distribution with a boxplot to see outliers
plot.sleepXmentalhealth <- ggplot(plot_data, aes(x = MENTALHEALTHSTATUS_T1, y = SLEEPSCORE)) +
  geom_boxplot(fill = "#009159") +
  geom_jitter(width = 0.2, alpha = 0.3) +
  ggtitle("Distribution of Muse S Sleep Score by Mental Health Status") +
  xlab("Mental Health Status") +
  ylab("Muse S Sleep Score") +
  theme_bw() +
  scale_x_discrete(
    labels = c(
      "1: Poor",
      "2: Fair",
      "3: Good",
      "4: Very Good",
      "5: Excellent"
    ),
    drop = FALSE
  ) +
  ylim(0, 100)

plot.sleepXmentalhealth
```




```
#source: https://ggplot2.tidyverse.org/reference/scale_discrete.html
#explanation: boxplot showing distribution of step counts with all health status categories inc
```

6 Model

6.1 Kruskal-Wallis Test for Health and Step Count

```
# kruskal-wallis test used for non-normal data and/or unequal group sizes comparing mean differ

library(dplyr)

# Prepare data
health_stepcount_data <- masterdata %>%
  filter(!is.na(HEALTHSTATUS_T1) & !is.na(STEPCOUNT))

# Perform Kruskal-Wallis test
kruskal.test(STEPCOUNT ~ HEALTHSTATUS_T1, data = health_stepcount_data)
```

```
    Kruskal-Wallis rank sum test

data:  STEPCOUNT by HEALTHSTATUS_T1
Kruskal-Wallis chi-squared = 9.5245, df = 3, p-value = 0.02307
```

```
# the p-value less than .05 demonstrates there are group differences

#source: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/kruskal.test
#explanation: non-parametric test for comparing multiple independent groups with unequal sample
```

6.2 Kruskal-Wallis Test for Health and Sleep Score

```
# kruskal-wallis test used for non-normal data and/or unequal group sizes comparing mean diffe

library(dplyr)

# Prepare data
health_sleepscore_data <- masterdata %>%
  filter(!is.na(HEALTHSTATUS_T1) & !is.na(SLEEPSCORE))

# Perform Kruskal-Wallis test
kruskal.test(SLEEPSCORE ~ HEALTHSTATUS_T1, data = health_sleepscore_data)
```

```

    Kruskal-Wallis rank sum test

data:  SLEEPSCORE by HEALTHSTATUS_T1
Kruskal-Wallis chi-squared = 4.9917, df = 3, p-value = 0.1724
```

```
# the p-value greater than .05 demonstrates there are no group differences

#source: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/kruskal.test
#explanation: non-parametric test for comparing multiple independent groups with unequal sample
```

6.3 Kruskal-Wallis Test for Mental Health and Step Count

```
# kruskal-wallis test used for non-normal data and/or unequal group sizes comparing mean diffe

library(dplyr)

# Prepare data
mentalhealth_stepcount_data <- masterdata %>%
  filter(!is.na(MENTALHEALTHSTATUS_T1) & !is.na(STEPCOUNT))

# Perform Kruskal-Wallis test
kruskal.test(STEPCOUNT ~ MENTALHEALTHSTATUS_T1, data = mentalhealth_stepcount_data)
```

```
    Kruskal-Wallis rank sum test

data:  STEPCOUNT by MENTALHEALTHSTATUS_T1
Kruskal-Wallis chi-squared = 8.2532, df = 4, p-value = 0.08273
```

```
# the p-value greater than .05 demonstrates there are no group differences

#source: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/kruskal.test
#explanation: non-parametric test for comparing multiple independent groups with unequal sample
```

6.4 Kruskal-Wallis Test for Mental Health and Sleep Score

```
# kruskal-wallis test used for non-normal data and/or unequal group sizes comparing mean differ

library(dplyr)

# Prepare data
mentalhealth_sleepscore_data <- masterdata %>%
  filter(!is.na(MENTALHEALTHSTATUS_T1) & !is.na(SLEEPSCORE))

# Perform Kruskal-Wallis test
kruskal.test(SLEEPSCORE ~ MENTALHEALTHSTATUS_T1, data = mentalhealth_sleepscore_data)
```

```
    Kruskal-Wallis rank sum test

data:  SLEEPSCORE by MENTALHEALTHSTATUS_T1
Kruskal-Wallis chi-squared = 3.8263, df = 4, p-value = 0.43
```

```
# the p-value greater than .05 demonstrates there are no group differences

#source: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/kruskal.test
#explanation: non-parametric test for comparing multiple independent groups with unequal sample
```

7 Creating Day vs. Sleep Score Scatter Plot

```
sleep_long <- masterdata %>%
  mutate(across(starts_with("SLEEPSCORE_T"), as.numeric)) %>%  # make all numeric
  pivot_longer(
    cols = starts_with("SLEEPSCORE_T"),
    names_to = "Day",
```

```
    values_to = "SleepScore"
  ) %>%
  mutate(Day = as.numeric(gsub("SLEEPSCORE_T", "", Day)))

library(ggplot2)

sleepscore_day <- ggplot(sleep_long, aes(x = Day, y = SleepScore)) +
  geom_point(color = "#4D1B7B") +
  stat_summary(fun = mean, geom = "line", color = "#A68DBD") +
  xlab("Night of Sleep") +
  ylab("MUSE S Sleep Score") +
  ggtitle("Average Sleep Score by Day") +
  theme_bw() +
  scale_x_continuous(
    breaks = 1:7,
    labels = c(
      "Sunday",
      "Monday",
      "Tuesday",
      "Wednesday",
      "Thursday",
      "Friday",
      "Saturday"
    )
  )

#source: https://ggplot2.tidyverse.org/reference/index.html, https://r4ds.hadley.nz/ (ch5-12)
#explanation: plot day x sleep score relationship as a scatterplot with a line representing the

# Print and save to the plots folder
print(sleepscore_day)
```

```
Warning: Removed 200 rows containing non-finite outside the scale range
(`stat_summary()`).
```

```
Warning: Removed 200 rows containing missing values or values outside the scale range
(`geom_point()`).
```

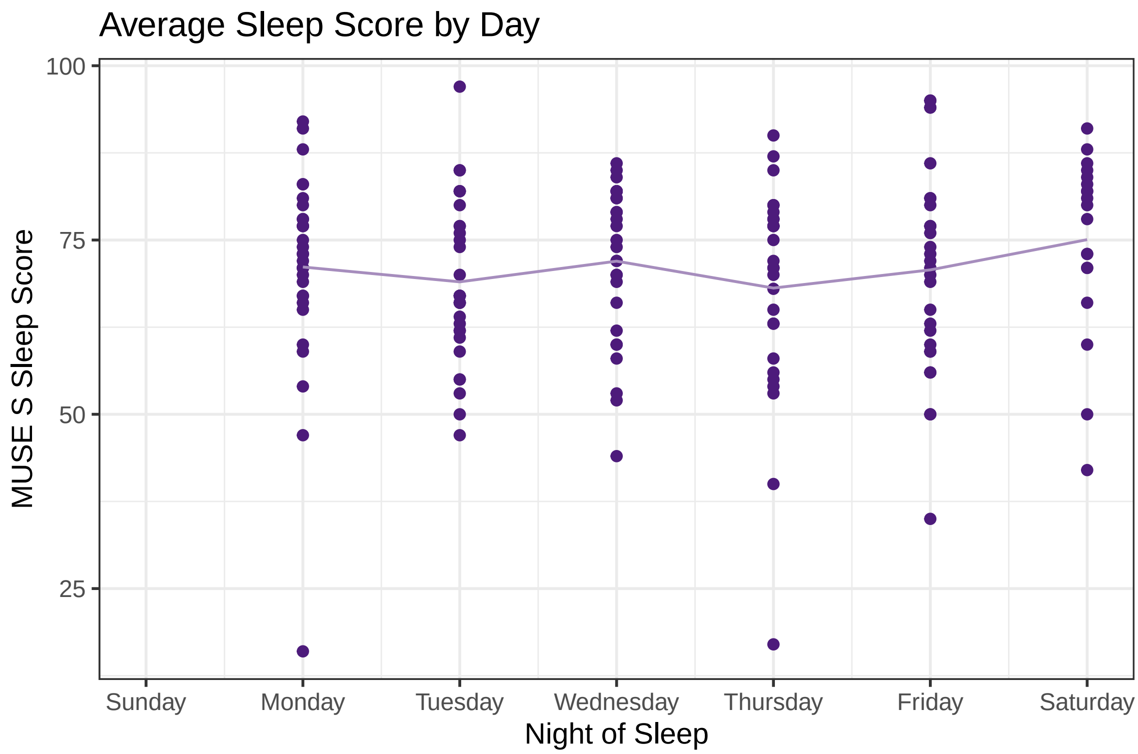


Figure 5: Figure 6. Average MUSE S Sleep Score Across Days of study participation. Sleep score was recorded on each day in reference to the score they achieved in their sleep from the night before. Average sleep scores remained relatively stable across Days 2 through 7 of data collection (participants did not wear their MUSE headbands before completing the first daily survey, so there is no data for Day 1). Although minor day-to-day fluctuations were observed, no clear upward or downward trend emerged in overall sleep quality."

```
#source: #source: Visualizing pre/post score (Sava, 2025): https://shanemccarty.github.io/FRIp
#explanation: shift to long format so thatday can be plotted on the x-axis
```

```
ggsave("plots/sleepscore_day.png",
       plot = sleepscore_day,
       width = 10, height = 8, dpi = 300)
```

```
Warning: Removed 200 rows containing non-finite outside the scale range
(`stat_summary()`).
```

```
Warning: Removed 200 rows containing missing values or values outside the scale range
(`geom_point()`).
```

```
#source: Visualize data in ggplot 2 (Silhavy & McCarty, 2025): https://shanemccarty.github.io/
#explanation: save the plot to the plots folder
```

8 Creating Day vs. Step Count

```
step_long <- masterdata %>%
  mutate(across(starts_with("STEPCOUNT_T"), as.numeric)) %>%  # make all numeric
  pivot_longer(
    cols = starts_with("STEPCOUNT_T"),
    names_to = "Day",
    values_to = "StepCount"
  ) %>%
  mutate(Day = as.numeric(gsub("STEPCOUNT_T", "", Day)))

library(ggplot2)

stepcount_day <- ggplot(step_long, aes(x = Day, y = StepCount)) +
  geom_point(color = "#cc397b") +
  stat_summary(fun = mean, geom = "line", color = "#f09cc1") +
  xlab("Day") +
  ylab("Fitbit Charge 6 Step Count") +
  ggtitle("Average Step Count by Day") +
  theme_bw() +
  scale_x_continuous(
    breaks = 1:7,
    labels = c(
      "Monday",
      "Tuesday",
      "Wednesday",
      "Thursday",
      "Friday",
      "Saturday",
      "Sunday"
    )
  )

#source: https://ggplot2.tidyverse.org/reference/index.html, https://r4ds.hadley.nz/ (ch5-12)
#explanation: plot day x step count relationship as a scatterplot with a line representing the

# Print and save to the plots folder
print(stepcount_day)
```

```
Warning: Removed 135 rows containing non-finite outside the scale range
(`stat_summary()`).
```

```
Warning: Removed 135 rows containing missing values or values outside the scale range
(`geom_point()`).
```

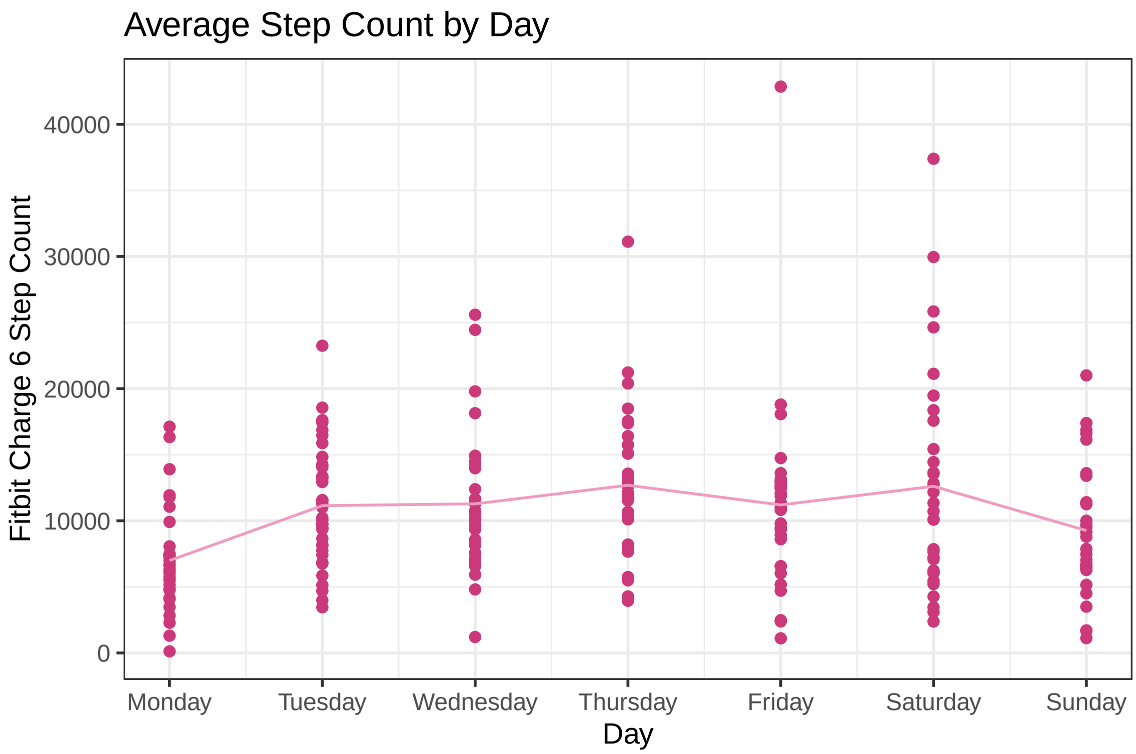


Figure 6: Figure 6. Average step count by day. Individual daily step counts recorded by the Fitbit Charge 6 are displayed for each day of the week. The pink line represents the mean step count per day, showing slightly higher averages midweek and on Saturday compared to Monday and Sunday.

```
#source: #source: Visualizing pre/post score (Sava, 2025): https://shanemccarty.github.io/FRIpl
#explanation: shift to long format so thatday can be plotted on the x-axis
```

```
ggsave("plots/stepcount_day.png",
       plot = stepcount_day,
       width = 10, height = 8, dpi = 300)
```

```
Warning: Removed 135 rows containing non-finite outside the scale range
(`stat_summary()`).
```

```
Warning: Removed 135 rows containing missing values or values outside the scale range
(`geom_point()`).
```

```
#source: Visualize data in ggplot 2 (Silhavy & McCarty, 2025): https://shanemccarty.github.io/F
#explanation: save the plot to the plots folder
```

9 Creating Step Count and Sleep Score Scatterplot

```
library(ggplot2)
sleepstep_scatter <- ggplot(masterdata, aes(STEPCOUNT,SLEEPSCORE)) +
  geom_smooth(method = "lm", color = "#cc397b") +
  geom_point(position = "jitter") +
  stat_summary(fun = mean, color = "#4d1b7b") +
  ylab("Muse S Sleep Score") +
  xlab("Step Count") +
  ggtitle("Sleep Score by Step Count") +
  theme_bw()

summary(masterdata$SLEEPSCORE)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  60.00   70.25   72.83   72.00   74.52   83.33
```

```
summary(masterdata$STEPCOUNT)
```

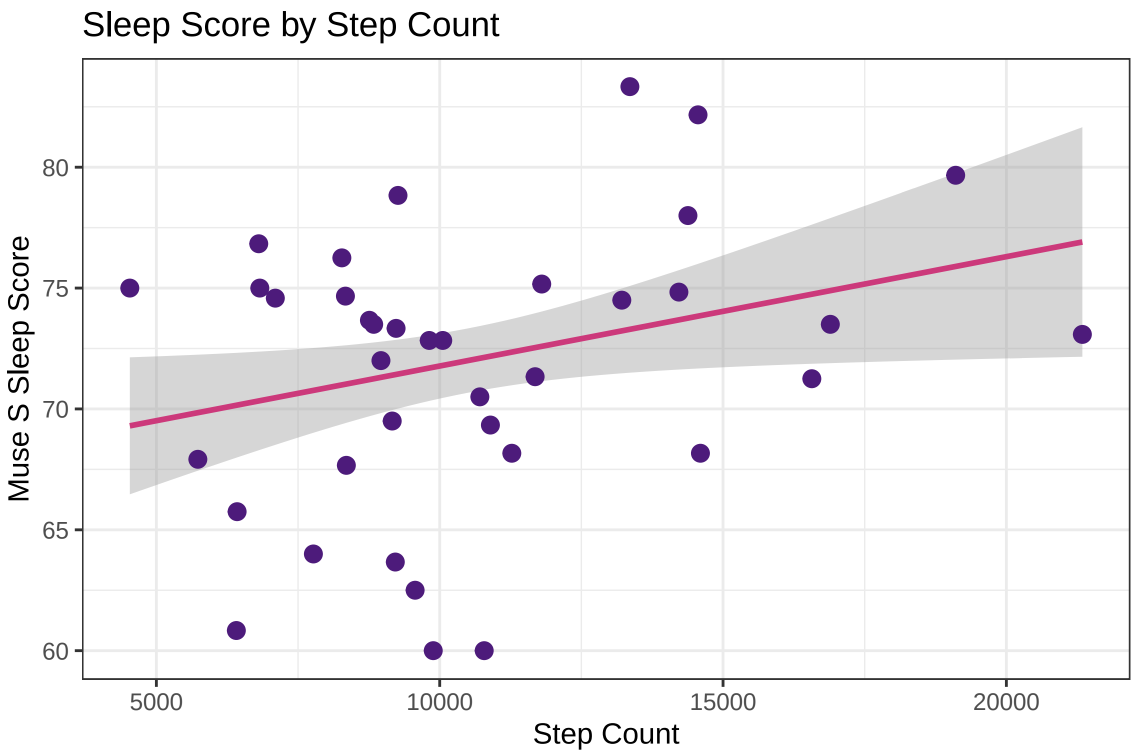
```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   4531    8931   10054   10489   10989   21341
```

```
lm(formula = SLEEPSCORE ~ STEPCOUNT, data = masterdata)
```

```
Call:
lm(formula = SLEEPSCORE ~ STEPCOUNT, data = masterdata)

Coefficients:
(Intercept)    STEPCOUNT
  6.725e+01    4.524e-04
```

```
print(sleepstep_scatter)
```

```
`geom_smooth()` using formula = 'y ~ x'

Warning: Removed 38 rows containing missing values or values outside the scale range
(`geom_segment()`).
```




```
#source: datacamp
```

```
ggsave("plots/sleepstep_scatter.png",
       plot = sleepstep_scatter,
       width = 10, height = 8, dpi = 300)
```

```
`geom_smooth()` using formula = 'y ~ x'
```

```
Warning: Removed 38 rows containing missing values or values outside the scale range
(`geom_segment()`).
```

```
#correlation_SC_vs_SS_spearman
cor.test(masterdata$STEPCOUNT, masterdata$SLEEPSCORE, method = 'spearman')
```

```
Warning in cor.test.default(masterdata$STEPCOUNT, masterdata$SLEEPSCORE, :
Cannot compute exact p-value with ties
```

```

    Spearman's rank correlation rho

data:  masterdata$STEPCOUNT and masterdata$SLEEPSCORE
S = 19730, p-value = 0.2639
alternative hypothesis: true rho is not equal to 0
```

```
sample estimates:
      rho
0.1577799
```

```
#source: https://www.r-bloggers.com/2021/10/pearson-correlation-in-r/, https://www.onlinespss.c
```

```
#correlation_SC_vs_SS_pearson
cor.test(masterdata$STEPCOUNT, masterdata$SLEEPSCORE, method = 'pearson')
```

```

	Pearson's product-moment correlation

data:  masterdata$STEPCOUNT and masterdata$SLEEPSCORE
t = 2.1637, df = 50, p-value = 0.03529
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.02141185 0.52368574
sample estimates:
      cor
0.2926024
```

```
#source: https://www.r-bloggers.com/2021/10/pearson-correlation-in-r/, https://www.onlinespss.c
```

9.1 *Means*

```
mean(masterdata$STEPCOUNT, na.rm = TRUE)
```

```
[1] 10489.42
```

```
mean(masterdata$SLEEPSCORE, na.rm = TRUE)
```

```
[1] 71.99679
```

9.2 *Medians*

```
#| label: Median-SleepandStep
median(masterdata$STEPCOUNT, na.rm = TRUE)
```

```
[1] 10053.5
```

```r
median(masterdata$SLEEPSCORE, na.rm = TRUE)
```

```
[1] 72.83333
```

9.3 *Standard Deviations*

```r
sd(masterdata$STEPCOUNT, na.rm = TRUE)
```

```
[1] 3200.734
```

```r
sd(masterdata$SLEEPSCORE, na.rm = TRUE)
```

```
[1] 4.948839
```

9.4 *Minimums and Maximums*

```r
min(masterdata$STEPCOUNT, na.rm = TRUE)
```

```
[1] 4531.286
```

```r
max(masterdata$STEPCOUNT, na.rm = TRUE)
```

```
[1] 21341
```

```r
min(masterdata$SLEEPSCORE, na.rm = TRUE)
```

```
[1] 60
```

```r
max(masterdata$SLEEPSCORE, na.rm = TRUE)
```

```
[1] 83.33333
```