Bella Beats case study

Emma Schenegg

2025-07-19

Introduction

Business Context

Bellabeat is a wellness technology company focused on creating smart products designed specifically for women. Their devices track physical activity, sleep, stress, and reproductive health to promote holistic wellbeing. Bellabeat emphasizes combining stylish design with health insights to support a balanced lifestyle. Their products serve as lifestyle partners, helping women monitor both mental and physical health. The brand is known for female-centric features and empowering users through personalized wellness data. Sršen, co-founder of Bellabeat, has requested an analysis of smart device usage data to gain insights into how consumers—especially women—use non-Bellabeat tracking devices. The objective is to understand key user behaviors, identify trending features, and explore opportunities to improve Bellabeat's product offerings and marketing strategy.

Guiding Problem

The primary problem identifying which trends are most relevant to Bellabeat's target audience. The goal is to reduce internal bias by examining the behaviours of users from competing products and using those findings to: • Improve Bellabeat's existing product features • Tailor marketing campaigns based on real consumer behaviour • Provide stakeholders with evidence-based recommendations for product development Stakeholders • Bellabeat co-founders • The marketing team • Product development and design teams

Datasets

The data used in this analysis comes from the publicly available FitBit Fitness Tracker Data provided by Möbius on Kaggle. It includes anonymized activity, sleep, heart rate, and weight data collected from 33 to 35 users over a period of 31 to 60 days between March and May 2016. Four key datasets were utilized: dailyActivity_merged.csv (tracking total steps, activity intensity levels, sedentary minutes, and calories burned), heartrate_seconds_merged.csv (second-by-second heart rate data in BPM), sleepDay_merged.csv (sleep duration, number of sleep records, and time spent in bed), and weightLogInfo_merged.csv (user weight in kilograms, BMI, and manual/automatic entry type). The activity, heart rate, and weight data were originally split across two separate datasets covering different time periods—12/03/2016 to 11/04/2016 and 12/04/2016 to 12/05/2016—and this analysis will later combine them using the rbind() function to ensure a continuous timeline for analysis. The sleep data is already consolidated and did not require merging.

Business Task Statement

To gain insights into trends in women's behavior with competitor smart tracking devices in order to guide Bellabeat's marketing strategy and product development, helping the brand better connect with existing and potential customers.

Install packages

```
options(repos = c(CRAN = "https://cloud.r-project.org"))

install.packages("tidyverse") # (ggplot2, dplyr,tidyr,readr, stringr etc...)
install.packages("lubridate") # Manipulation of date/time
install.packages("corrplot")  # Correlation heat map
```

Load packages

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.5.1
```

```
## Warning: package 'tidyr' was built under R version 4.5.1
```

```
## Warning: package 'dplyr' was built under R version 4.5.1
```

```
## Warning: package 'lubridate' was built under R version 4.5.1
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.2     ✓ tibble    3.2.1
## ✓ lubridate 1.9.4     ✓ tidyr     1.3.1
## ✓ purrr     1.0.4
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to be
come errors
```

```r
library(lubridate)
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.5.1
```

```
## corrplot 0.95 loaded
```

Bind and load data sets

```
# Using rbind function to link the matching data from the 12/03/16-11/04/16 (dd/mm/yyyy) peri
od to the 12/04/16-12/05/2016 period since they are located in different data sets.

Activity <- rbind(read.csv("C:/Users/emma3/Downloads/Fitabase Data 4.12.16-5.12.16/dailyActiv
ity_merged2.csv"),read.csv("C:/Users/emma3/Downloads/mturkfitbit_export_3.12.16-4.11.16/daily
Activity_merged.csv"))

Weight <- rbind(read.csv("C:/Users/emma3/Downloads/Fitabase Data 4.12.16-5.12.16/weightLogInf
o_merged.csv"),read.csv("C:/Users/emma3/Downloads/mturkfitbit_export_3.12.16-4.11.16/weightLo
gInfo_merged1.csv"))

Heartrate <- rbind(read.csv("C:/Users/emma3/Downloads/Fitabase Data 4.12.16-5.12.16/heartrate
_seconds_merged.csv"), read.csv("C:/Users/emma3/Downloads/mturkfitbit_export_3.12.16-4.11.16/
heartrate_seconds_merged1.csv"))

Sleep <- read.csv("C:/Users/emma3/Downloads/Fitabase Data 4.12.16-5.12.16/sleepDay_merged.cs
v")
 #There is only one Data set containing sleep data, so there is no need to use rbind functio
n.
```

Visualise Data

```
# Using head function to check columns and have a visuals on the data.

head(Activity)
```

```
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
## 1 1503960366    4/12/2016      13162          8.50            8.50
## 2 1503960366    4/13/2016      10735          6.97            6.97
## 3 1503960366    4/14/2016      10460          6.74            6.74
## 4 1503960366    4/15/2016       9762          6.28            6.28
## 5 1503960366    4/16/2016      12669          8.16            8.16
## 6 1503960366    4/17/2016       9705          6.48            6.48
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               1.88                     0.55
## 2                        0               1.57                     0.69
## 3                        0               2.44                     0.40
## 4                        0               2.14                     1.26
## 5                        0               2.71                     0.41
## 6                        0               3.19                     0.78
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                6.06                       0                25
## 2                4.71                       0                21
## 3                3.91                       0                30
## 4                2.83                       0                29
## 5                5.04                       0                36
## 6                2.51                       0                38
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories
## 1                  13                  328              728     1985
## 2                  19                  217              776     1797
## 3                  11                  181             1218     1776
## 4                  34                  209              726     1745
## 5                  10                  221              773     1863
## 6                  20                  164              539     1728
```

```
head(Weight)
```

```
##           Id                  Date WeightKg WeightPounds Fat   BMI
## 1 1503960366  5/2/2016 11:59:59 PM     52.6     115.9631  22 22.65
## 2 1503960366  5/3/2016 11:59:59 PM     52.6     115.9631  NA 22.65
## 3 1927972279  4/13/2016 1:08:52 AM    133.5     294.3171  NA 47.54
## 4 2873212765 4/21/2016 11:59:59 PM     56.7     125.0021  NA 21.45
## 5 2873212765 5/12/2016 11:59:59 PM     57.3     126.3249  NA 21.69
## 6 4319703577 4/17/2016 11:59:59 PM     72.4     159.6147  25 27.45
##   IsManualReport        LogId
## 1           True 1.462234e+12
## 2           True 1.462320e+12
## 3          False 1.460510e+12
## 4           True 1.461283e+12
## 5           True 1.463098e+12
## 6           True 1.460938e+12
```

```
head(Heartrate)
```

```
##           Id                 Time Value
## 1 2022484408 4/12/2016 7:21:00 AM    97
## 2 2022484408 4/12/2016 7:21:05 AM   102
## 3 2022484408 4/12/2016 7:21:10 AM   105
## 4 2022484408 4/12/2016 7:21:20 AM   103
## 5 2022484408 4/12/2016 7:21:25 AM   101
## 6 2022484408 4/12/2016 7:22:05 AM    95
```

```
head(Sleep)
```

```
##           Id              SleepDay TotalSleepRecords TotalMinutesAsleep
## 1 1503960366 4/12/2016 12:00:00 AM                 1                327
## 2 1503960366 4/13/2016 12:00:00 AM                 2                384
## 3 1503960366 4/15/2016 12:00:00 AM                 1                412
## 4 1503960366 4/16/2016 12:00:00 AM                 2                340
## 5 1503960366 4/17/2016 12:00:00 AM                 1                700
## 6 1503960366 4/19/2016 12:00:00 AM                 1                304
##   TotalTimeInBed
## 1            346
## 2            407
## 3            442
## 4            367
## 5            712
## 6            320
```

Inspecting Data Formats

```
# Using the str function to see if the formatting of columns are consistent and correct.

str(Activity)
```

```
## 'data.frame':    1397 obs. of  15 variables:
##  $ Id                      : num  1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityDate            : chr  "4/12/2016" "4/13/2016" "4/14/2016" "4/15/2016" ...
##  $ TotalSteps              : int  13162 10735 10460 9762 12669 9705 13019 15506 10544 9819
...
##  $ TotalDistance           : num  8.5 6.97 6.74 6.28 8.16 ...
##  $ TrackerDistance         : num  8.5 6.97 6.74 6.28 8.16 ...
##  $ LoggedActivitiesDistance: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveDistance      : num  1.88 1.57 2.44 2.14 2.71 ...
##  $ ModeratelyActiveDistance: num  0.55 0.69 0.4 1.26 0.41 ...
##  $ LightActiveDistance     : num  6.06 4.71 3.91 2.83 5.04 ...
##  $ SedentaryActiveDistance : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveMinutes       : int  25 21 30 29 36 38 42 50 28 19 ...
##  $ FairlyActiveMinutes     : int  13 19 11 34 10 20 16 31 12 8 ...
##  $ LightlyActiveMinutes    : int  328 217 181 209 221 164 233 264 205 211 ...
##  $ SedentaryMinutes        : int  728 776 1218 726 773 539 1149 775 818 838 ...
##  $ Calories                : int  1985 1797 1776 1745 1863 1728 1921 2035 1786 1775 ...
```

```
str(Weight)
```

```
## 'data.frame':    100 obs. of  8 variables:
##  $ Id            : num  1.50e+09 1.50e+09 1.93e+09 2.87e+09 2.87e+09 ...
##  $ Date          : chr  "5/2/2016 11:59:59 PM" "5/3/2016 11:59:59 PM" "4/13/2016 1:08:52 A
M" "4/21/2016 11:59:59 PM" ...
##  $ WeightKg      : num  52.6 52.6 133.5 56.7 57.3 ...
##  $ WeightPounds  : num  116 116 294 125 126 ...
##  $ Fat           : int  22 NA NA NA NA 25 NA NA NA NA ...
##  $ BMI           : num  22.6 22.6 47.5 21.5 21.7 ...
##  $ IsManualReport: chr  "True" "True" "False" "True" ...
##  $ LogId         : num  1.46e+12 1.46e+12 1.46e+12 1.46e+12 1.46e+12 ...
```

```
str(Heartrate)
```

```
## 'data.frame':    3638339 obs. of  3 variables:
##  $ Id   : num  2.02e+09 2.02e+09 2.02e+09 2.02e+09 2.02e+09 ...
##  $ Time : chr  "4/12/2016 7:21:00 AM" "4/12/2016 7:21:05 AM" "4/12/2016 7:21:10 AM" "4/12/
2016 7:21:20 AM" ...
##  $ Value: int  97 102 105 103 101 95 91 93 94 93 ...
```

```
str(sleep)
```

```
## 'data.frame':    20 obs. of  3 variables:
##  $ extra: num  0.7 -1.6 -0.2 -1.2 -0.1 3.4 3.7 0.8 0 2 ...
##  $ group: Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ ID   : Factor w/ 10 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
```

```
#Date and time are in characters rather than date format, time indicate AM/PM rather than 23:
59:59 format. Date is in US style (mm/dd/yyyy) rather than in uk style (dd/mm/yyyy). Activity
does not present time but only date, Sleep presents time but all indicate 00:00:00.
```

Adjusting Dates

Dates format

```
# Using POSIXct to indicate the curing character string format in date format.
# Using Sis.Timezone to match the timezone data times are actually in
# Formatting time in a UK format dd/mm/yyy HH:MM:SS for easier understanding and avoid any fu
ture confusion and saving it as "FormattedTime".

Activity$ActivityDate <- as.POSIXct(Activity$ActivityDate, format="%m/%d/%Y", tz= Sys.timezon
e())
Activity$FormattedTime <- format(Activity$ActivityDate, format = "%d/%m/%Y")
```

```
# Using format %I/%M/%S %p because 12h format (AM/PM) for those data sets.

Weight$Date <- as.POSIXct(Weight$Date, format="%m/%d/%Y %I:%M:%S %p", tz= Sys.timezone())
Weight$FormattedTime <- format(Weight$Date, format = "%d/%m/%Y %H:%M:%S")

Heartrate$Time <- as.POSIXct(Heartrate$Time, format="%m/%d/%Y %I:%M:%S %p", tz= Sys.timezone
())
Heartrate$FormattedTime <- format(Heartrate$Time, format = "%d/%m/%Y %H:%M:%S")

Sleep$SleepDay <- as.POSIXct(Sleep$SleepDay, format="%m/%d/%Y %H:%M:%S", tz= Sys.timezone())
Sleep$FormattedTime <- format(Sleep$SleepDay, format = "%d/%m/%Y") # Since Sleep Data set tim
e are all 00:00:00, only the date is relevant and kept.
```

Remove previous date variable

```
# Remove previous US Date/time columns.

Activity <- Activity %>% select(-ActivityDate)
Weight <- Weight %>%  select(-Date)
Heartrate <- Heartrate %>% select(-Time)
Sleep <- Sleep %>% select(-SleepDay)
```

Distinct values

```
# Using n_distinct function to summarise the number of unique participants (Id) in each data
set.

n_distinct(Activity$Id)          #35
```

```
## [1] 35
```

```
n_distinct(Weight$Id)            #13
```

```
## [1] 13
```

```
n_distinct(Heartrate$Id)         #15
```

```
## [1] 15
```

```
n_distinct(Sleep$Id)             #24
```

```
## [1] 24
```

Mean steps

```
# Check if users meets in average the recommended 10 000 steps a day (Tudor-Locke & Bassett,
2004).

mean(Activity$TotalSteps, na.rm = TRUE)
```

```
## [1] 7280.898
```

```
# Average steps = 7281, the mean doesn't reach daily steps recommendation
```

Creating New Variables

Total Active Minutes

```
# Create a column for total daily active minutes to group Lightly Active, Fairly Active and V
ery Active minutes.

Activity <- Activity %>%
  mutate(TotalActiveMinutes = LightlyActiveMinutes + FairlyActiveMinutes + VeryActiveMinutes)
```

Awake Sedentary Time

```
# Create a column for sedentary time that doesn't take into account sleeping time and where s
leeping time higher than sedentary time is not taken into account (After checking Sedentary a
wake time values some are negative and cannot be accounted as not accurate since total minute
s asleep cannot be more than sedentary time).

Activity$FormattedTime <- dmy(Activity$FormattedTime) # parse the datetime columns
Sleep$FormattedTime <- dmy(Sleep$FormattedTime)

Activity %>% count(Id, FormattedTime) %>% filter(n > 1) # check if id/formatted time have uni
que values
```

```
##             Id FormattedTime n
## 1  1503960366    2016-04-12 2
## 2  1624580081    2016-04-12 2
## 3  1844505072    2016-04-12 2
## 4  1927972279    2016-04-12 2
## 5  2022484408    2016-04-12 2
## 6  2026352035    2016-04-12 2
## 7  2320127002    2016-04-12 2
## 8  2347167796    2016-04-12 2
## 9  2873212765    2016-04-12 2
## 10 3977333714    2016-04-12 2
## 11 4020332650    2016-04-12 2
## 12 4057192912    2016-04-12 2
## 13 4445114986    2016-04-12 2
## 14 4558609924    2016-04-12 2
## 15 4702921684    2016-04-12 2
## 16 5553957443    2016-04-12 2
## 17 6962181067    2016-04-12 2
## 18 7007744171    2016-04-12 2
## 19 7086361926    2016-04-12 2
## 20 8053475328    2016-04-12 2
## 21 8253242879    2016-04-12 2
## 22 8378563200    2016-04-12 2
## 23 8792009665    2016-04-12 2
## 24 8877689391    2016-04-12 2
```

```
Sleep %>% count(Id, FormattedTime) %>% filter(n > 1)
```

```
##            Id FormattedTime n
## 1 4388161847    2016-05-05 2
## 2 4702921684    2016-05-07 2
## 3 8378563200    2016-04-25 2
```

```
Sleep_unique <- Sleep %>%
  distinct(Id, FormattedTime, .keep_all = TRUE) # remove duplicates

Activity_unique <-Activity %>%
  distinct(Id, FormattedTime, .keep_all = TRUE)

Activity <- Activity_unique %>%   # join Sleeping time and Sedentary time that belong to two
different data frames
  full_join(
    Sleep_unique[, c("Id", "FormattedTime", "TotalMinutesAsleep")],
    by = c("Id", "FormattedTime") # group new column by Id and date
  ) %>%
  arrange(Id, FormattedTime) %>%
  mutate(
    SedentaryAwakeTime = ifelse(
      SedentaryMinutes > TotalMinutesAsleep,
      SedentaryMinutes - TotalMinutesAsleep, NA)) # only keep values where Sedentary > Sleep,
else NA
# value_if_true, value_if_false (NA)
```

Time In Bed Awake

```
# Create column for time spent in bed awake by subtracting Sleeping time to Time in bed.

Sleep$TimeInBedAwake <- Sleep$TotalTimeInBed - Sleep$TotalMinutesAsleep
```

Means

Sleep

```
Sleep %>%
  select(TimeInBedAwake, TotalMinutesAsleep, TotalSleepRecords) %>%
  summary()
```

```
##  TimeInBedAwake   TotalMinutesAsleep TotalSleepRecords
##  Min.   :  0.00   Min.   : 58.0      Min.   :1.000
##  1st Qu.: 17.00   1st Qu.:361.0      1st Qu.:1.000
##  Median : 25.00   Median :433.0      Median :1.000
##  Mean   : 39.17   Mean   :419.5      Mean   :1.119
##  3rd Qu.: 40.00   3rd Qu.:490.0      3rd Qu.:1.000
##  Max.   :371.00   Max.   :796.0      Max.   :3.000
```

Activity

```
Activity %>%
  select(TotalSteps, VeryActiveMinutes, FairlyActiveMinutes, LightlyActiveMinutes, TotalActiv
eMinutes, SedentaryMinutes) %>%
  summary()
```

```
##    TotalSteps    VeryActiveMinutes FairlyActiveMinutes LightlyActiveMinutes
##  Min.   :    0   Min.   :  0.00    Min.   :  0.0       Min.   :  0.0
##  1st Qu.: 3321   1st Qu.:  0.00    1st Qu.:  0.0       1st Qu.:117.0
##  Median : 7142   Median :  2.00    Median :  6.0       Median :196.0
##  Mean   : 7377   Mean   : 19.87    Mean   : 13.6       Mean   :188.1
##  3rd Qu.:10645   3rd Qu.: 30.00    3rd Qu.: 18.0       3rd Qu.:263.0
##  Max.   :36019   Max.   :210.00    Max.   :660.0       Max.   :720.0
##  TotalActiveMinutes SedentaryMinutes
##  Min.   :  0.0      Min.   :   0
##  1st Qu.:135.0      1st Qu.: 734
##  Median :244.0      Median :1062
##  Mean   :221.6      Mean   :1001
##  3rd Qu.:315.0      3rd Qu.:1246
##  Max.   :720.0      Max.   :1440
```

Weight

```
Weight %>%
  select(BMI, WeightKg) %>%
  summary()
```

```
##       BMI           WeightKg    
##  Min.   :21.45   Min.   : 52.6  
##  1st Qu.:24.00   1st Qu.: 61.5  
##  Median :24.39   Median : 62.5  
##  Mean   :25.37   Mean   : 72.5  
##  3rd Qu.:25.59   3rd Qu.: 85.3  
##  Max.   :47.54   Max.   :133.5
```

Heart rate

```
Heartrate %>%
  select(Value) %>%
  summary()
```

```
##      Value      
##  Min.   : 36.0  
##  1st Qu.: 64.0  
##  Median : 74.0  
##  Mean   : 78.1  
##  3rd Qu.: 88.0  
##  Max.   :203.0
```

Weekend vs Weekdays breakdown

Activity

```
Activity$weekday <- weekdays(Activity$FormattedTime)
Activity$day_type <- ifelse(Activity$weekday %in% c("Saturday", "Sunday"), "Weekend", "Weekda
y")

Activity %>%
  group_by(day_type) %>%
  summarise(
    avg_steps = mean(TotalSteps, na.rm = TRUE),
    avg_sedentary = mean(SedentaryAwakeTime, na.rm = TRUE),
    avg_TotalActiveMinutes = mean(TotalActiveMinutes, na.rm = TRUE),
    avg_Calories = mean(Calories, na.rm = TRUE)
  )
```

```
## # A tibble: 2 × 5
##   day_type avg_steps avg_sedentary avg_TotalActiveMinutes avg_Calories
##   <chr>        <dbl>         <dbl>                  <dbl>        <dbl>
## 1 Weekday      7453.          347.                   223.        2302.
## 2 Weekend      7188.          332.                   219.        2277.
```

```
# Slightly more active and less sedentary on the weekend but not a big difference.
```

Sleep

```r
Sleep$weekday <- weekdays(Sleep$FormattedTime)
Sleep$day_type <- ifelse(Sleep$weekday %in% c("Saturday", "Sunday"), "Weekend", "Weekday")

Sleep %>%
  group_by(day_type) %>%
  summarise(
    avg_TotalMinutesAsleep = mean(TotalMinutesAsleep, na.rm = TRUE),
    avg_TimeInBedAwake = mean(TimeInBedAwake, na.rm = TRUE)
  )
```

```
## # A tibble: 2 × 3
##   day_type avg_TotalMinutesAsleep avg_TimeInBedAwake
##   <chr>                     <dbl>              <dbl>
## 1 Weekday                    413.               36.8
## 2 Weekend                    436.               45.5
```

```r
# People spend more time sleeping on the weekend.
```

Heart Rate

```r
Heartrate$FormattedTime <- dmy_hms(Heartrate$FormattedTime)
Heartrate$weekday <- weekdays(Heartrate$FormattedTime)
Heartrate$day_type <- ifelse(Heartrate$weekday %in% c("Saturday", "Sunday"), "Weekend", "Week
day")

Heartrate %>%
  group_by(day_type) %>%
  summarise(
    avg_Value = mean(Value, na.rm = TRUE)
  )
```

```
## # A tibble: 2 × 2
##   day_type avg_Value
##   <chr>        <dbl>
## 1 Weekday       77.8
## 2 Weekend       78.9
```

```r
# Heart Rate is very slightly higher on the weekend
```

Weight

```
Weight$FormattedTime <- dmy_hms(Weight$FormattedTime)
Weight$weekday <- weekdays(Weight$FormattedTime)
Weight$day_type <- ifelse(Weight$weekday %in% c("Saturday", "Sunday"), "Weekend", "Weekday")

Weight %>%
  group_by(day_type) %>%
  summarise(
    avg_WeightKg = mean(WeightKg, na.rm = TRUE),
    avg_BMI = mean(BMI, na.rn= TRUE)
  )
```

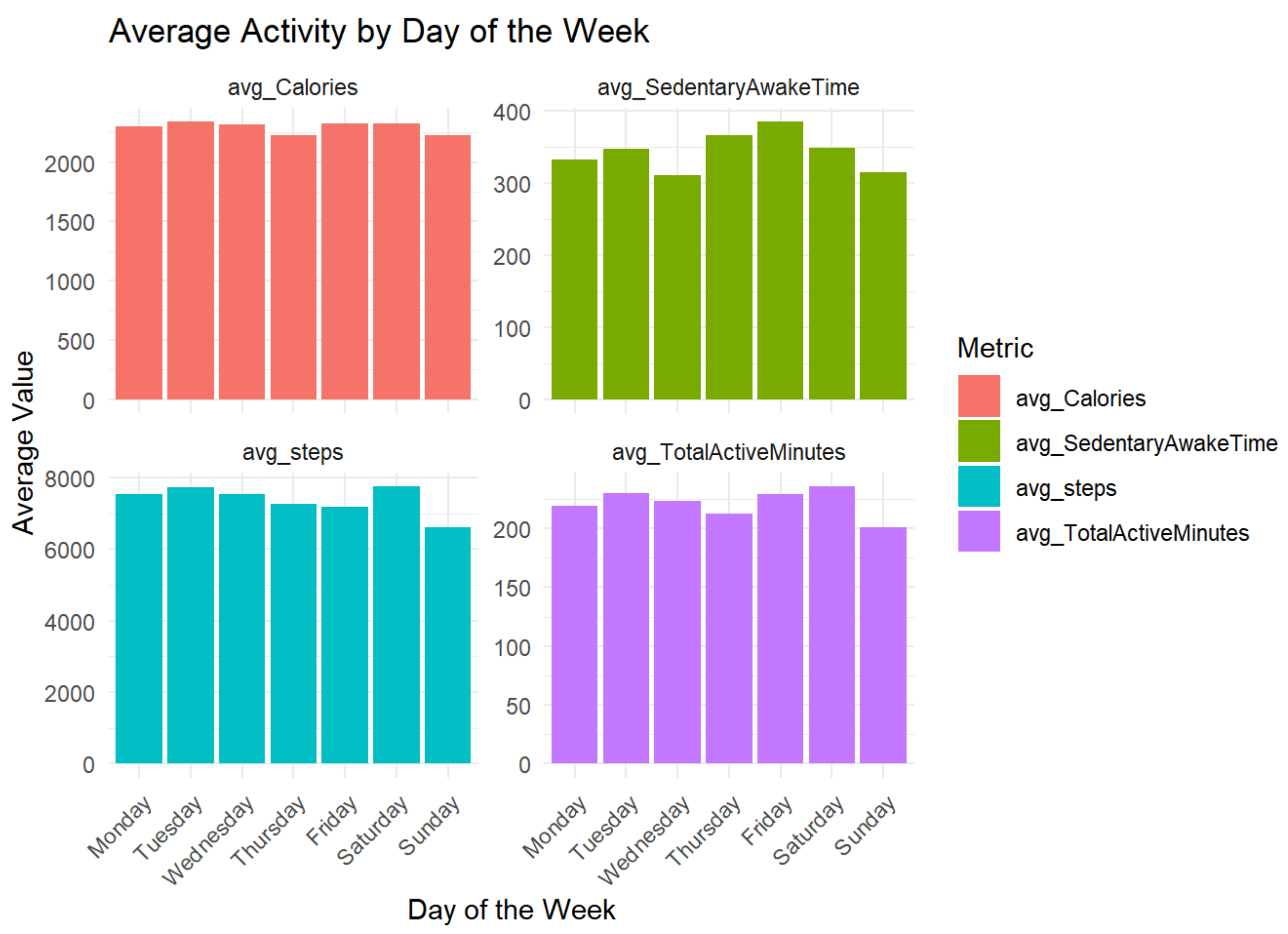
```
## # A tibble: 2 × 3
##   day_type avg_WeightKg avg_BMI
##   <chr>           <dbl>   <dbl>
## 1 Weekday          72.2    25.2
## 2 Weekend          73.5    25.9
```

```
# Weight and BMI higher on Weekends.
```

Daily Activity Breakdown

Activity

```r
summary_weekday <- Activity %>%
  group_by(weekday) %>%
  summarise(
    avg_steps = mean(TotalSteps, na.rm = TRUE),
    avg_SedentaryAwakeTime = mean(SedentaryAwakeTime, na.rm = TRUE),
    avg_TotalActiveMinutes = mean(TotalActiveMinutes, na.rm = TRUE),
    avg_Calories = mean(Calories, na.rm = TRUE)
  ) %>%
  mutate(weekday = factor(weekday,levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "F
riday", "Saturday", "Sunday"))) # order weekdays Monday to Sunday order rather than alphabeti
cally (mutate).


plot_data <- summary_weekday %>%
  pivot_longer(cols = starts_with("avg_"),
               names_to = "Metric",
               values_to = "Value")

ggplot(plot_data, aes(x = weekday, y = Value, fill = Metric)) +
  geom_col(position = "dodge") +
  facet_wrap(~ Metric, scales = "free_y") +  # separate y-axis scales per metric
  labs(
    title = "Average Activity by Day of the Week",
    x = "Day of the Week",
    y = "Average Value",
    fill = "Metric"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

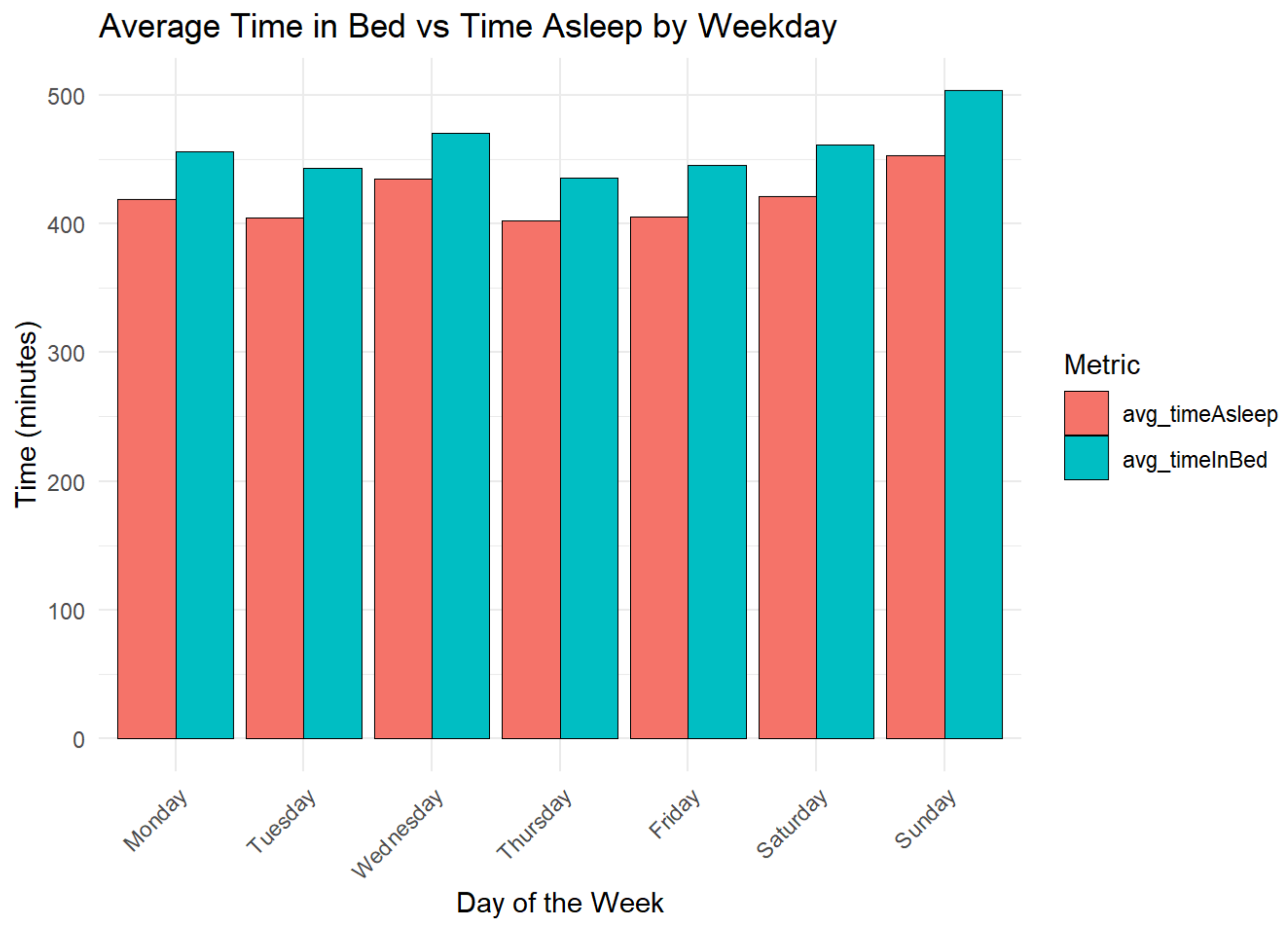


```
# People are less active on Friday and Sunday but most active on Saturday.
```

Sleep

```
summary_sleep <- Sleep %>%
  group_by(weekday) %>%
  summarise(
    avg_timeAsleep = mean(TotalMinutesAsleep, na.rm = TRUE),
    avg_timeInBed = mean(TotalTimeInBed, na.rm = TRUE),
  ) %>%
  mutate(weekday = factor(weekday,levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "F
riday", "Saturday", "Sunday")))

plot_sleep <- summary_sleep %>%
  pivot_longer(cols = starts_with("avg_"),
               names_to = "Metric",
               values_to = "Value")

ggplot(plot_sleep, aes(x = weekday, y = Value, fill = Metric)) +
  geom_col(position = "dodge",
           color = "black",
           size = 0.1
  ) +
  labs(
    title = "Average Time in Bed vs Time Asleep by Weekday",
    x = "Day of the Week",
    y = "Time (minutes)",
    fill = "Metric"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

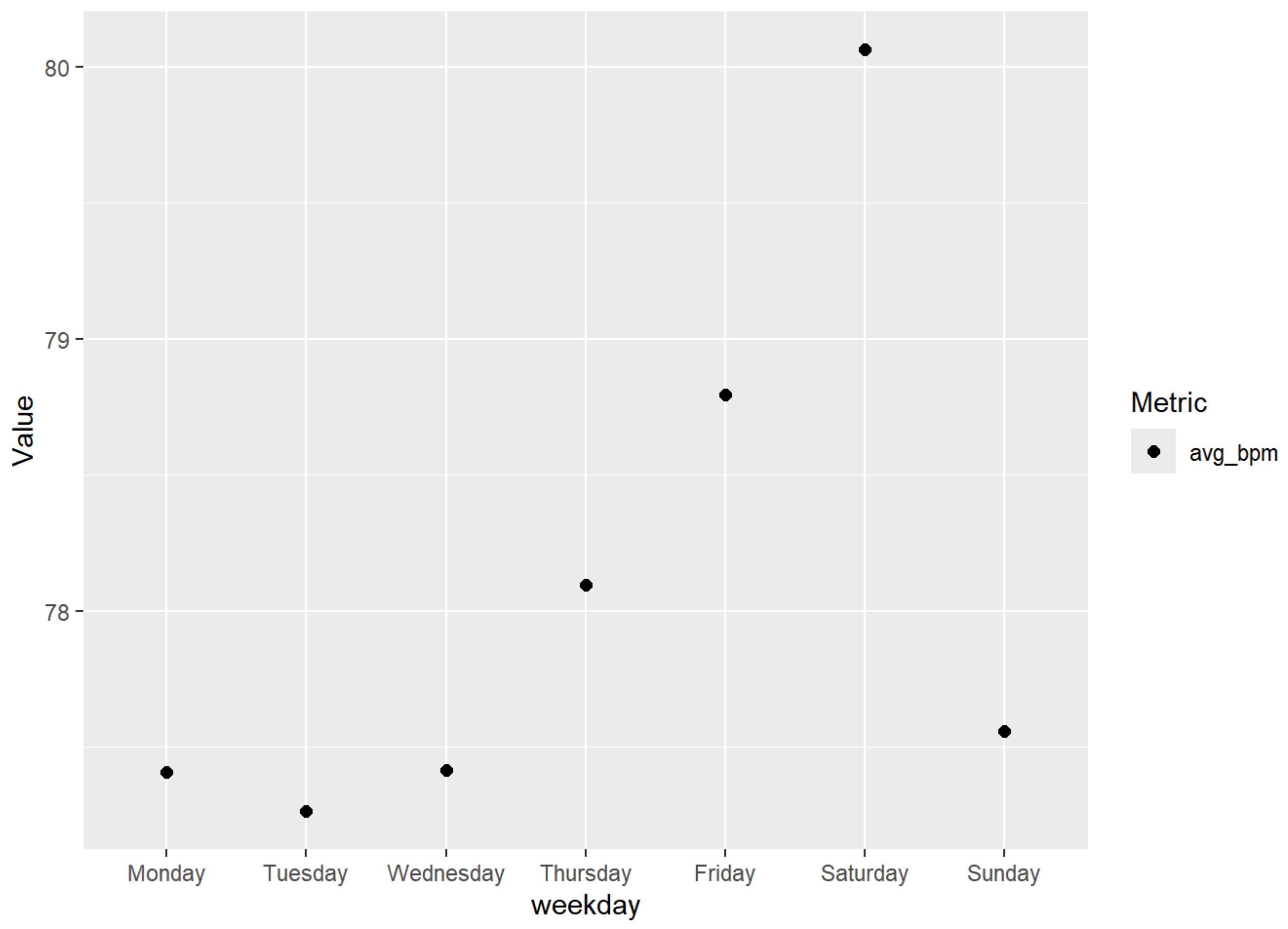


```
# People spend most time asleep and in bed on Sunday and Wednesday and less time both asleep
and in bed on Thursday, Tuesday and Friday.
```

Heart Rate

```
summary_heartrate <- Heartrate %>%
  group_by(weekday) %>%
  summarise(
    avg_bpm = mean(Value, na.rm = TRUE),
  ) %>%
  mutate(weekday = factor(weekday,levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "F
riday", "Saturday", "Sunday")))

plot_heartrate <- summary_heartrate %>%
  pivot_longer(cols = starts_with("avg_"),
               names_to = "Metric",
               values_to = "Value")

ggplot(plot_heartrate, aes(x = weekday, y = Value, fill = Metric)) +
  geom_line(size = 1.2)+
  geom_point(size = 2)
```

```
## `geom_line()`: Each group consists of only one observation.
## ℹ Do you need to adjust the group aesthetic?
```



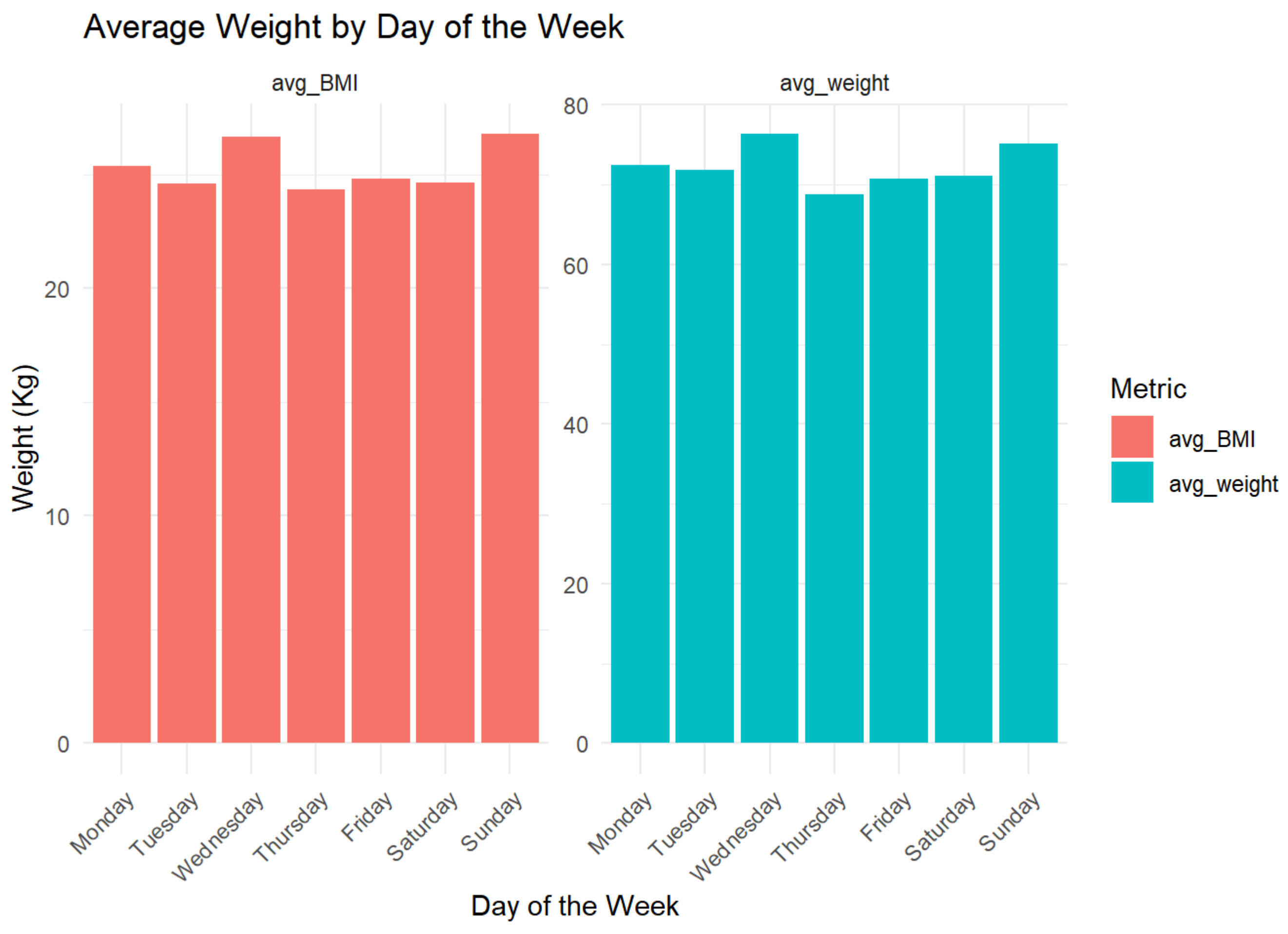
```
labs(
  title = "Average Heart rate by Weekday",
  x = "Day of the Week",
  y = "Heartrate (BPM)",
  fill = "Metric"
) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## NULL
```

```
# BPM is very slightly higher on Saturday and lower on Monday, Tuesday and Wednesday.
```

Weight

```
summary_weight <- Weight%>%
  group_by(weekday) %>%
  summarise(
    avg_weight = mean(WeightKg, na.rm = TRUE),
    avg_BMI = mean(BMI, na.rm = TRUE),
  ) %>%
  mutate(weekday = factor(weekday,levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "F
riday", "Saturday", "Sunday")))

plot_weight <- summary_weight %>%
  pivot_longer(cols = starts_with("avg_"),
               names_to = "Metric",
               values_to = "Value")

ggplot(plot_weight, aes(x = weekday, y = Value, fill = Metric)) +
  geom_col() +
  facet_wrap(~ Metric, scales = "free_y") +
  labs(
    title = "Average Weight by Day of the Week",
    x = "Day of the Week",
    y = "Weight (Kg)",
    fill = "Metric"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



```
# Both weight and BMI are higher on Wednesday and Sunday and slightly lower on Thursday.
```

Correlations

```
# Create a common column name "Date" for each data frame that only use d/m/y and no h:m:s in
order to use it for correlation within all data frame as some data frames use h/m/s and other
don't.

Activity$Date <- as.Date(Activity$FormattedTime, format = "%d/%m/%Y")
Sleep$Date <- as.Date (Sleep$FormattedTime, format = "%d/%m/%Y")
Heartrate$Date <- as.Date(Heartrate$FormattedTime, format = "%d/%m/%Y %H:%M:%S")
Weight$Date <- as.Date(Weight$FormattedTime, format = "%d/%m/%Y %H:%M:%S")
```

```
# There is too many Heart rate data as data have been recorded every 5minutes of each date. N
ext step will aggregate the Heart Rate data by computing the daily average, in order to reduc
e multiple values per day and match them with the corresponding single daily record from othe
r data frames.

Heartrate_daily <- Heartrate %>%
  group_by(Id, Date) %>%
  summarize(AvgHeartRate = mean(Value, na.rm = TRUE))
```

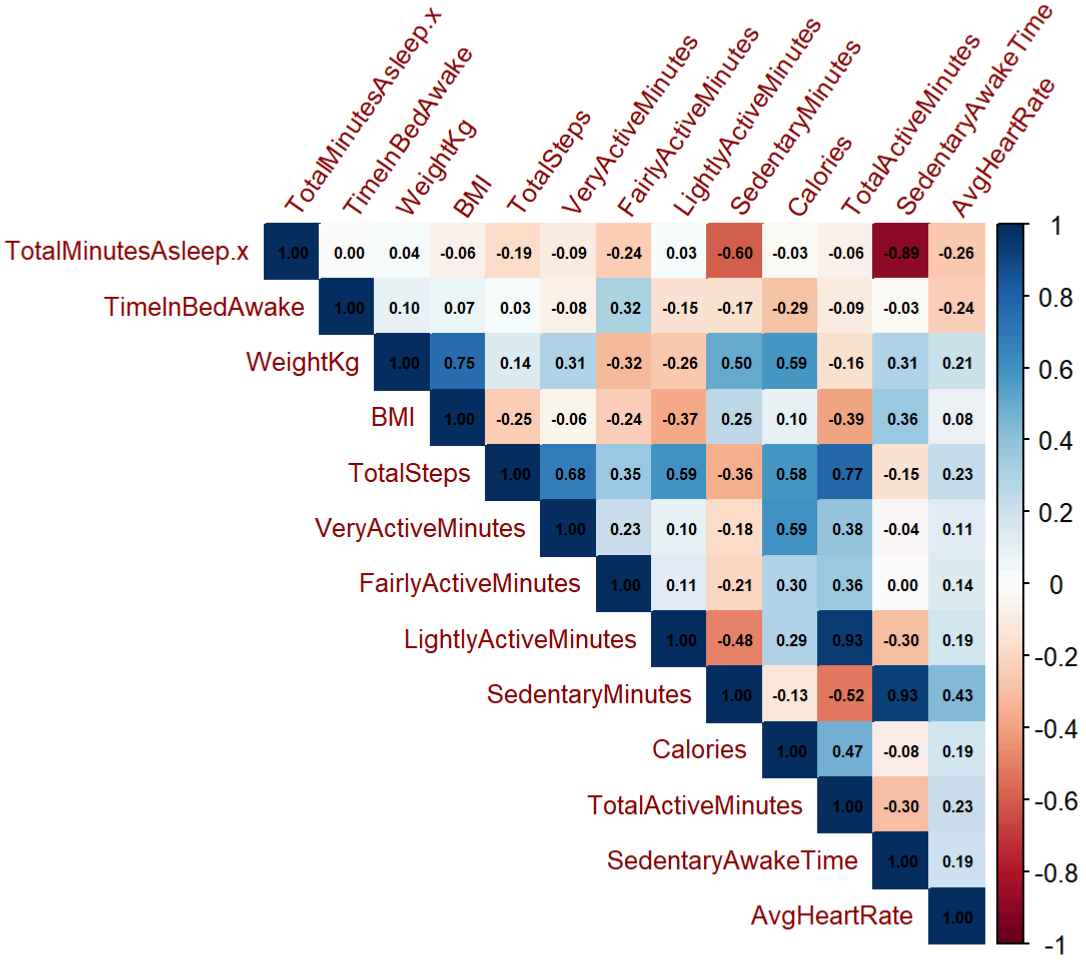
```
## `summarise()` has grouped output by 'Id'. You can override using the `.groups`
## argument.
```

Multi correlation matrix

```r
# Creating a data frame that merges all 4 main data frames by Id and Date.

Allmerged <- list(Sleep, Weight, Activity, Heartrate_daily) %>%
  reduce(full_join, by = c("Id", "Date"))

# Before to run correlation matrix. Remove unneeded variable for more clarity.

Allmerged <- Allmerged %>% select(-Id, -Fat, -TotalDistance, -TotalSleepRecords, -TotalMinute
sAsleep.y, -WeightPounds, -TotalTimeInBed, - LogId, -LoggedActivitiesDistance, -TrackerDistan
ce, -VeryActiveDistance, -ModeratelyActiveDistance, -LightActiveDistance, -SedentaryActiveDis
tance)


# Select only the numeric columns to avoid errors.

numeric_data <- Allmerged %>% select(where(is.numeric))


# To avoid error message, checks each numeric column to see if its standard deviation is 0 (a
ll values are the same) and stores the result as a logical vector.

zero_var_cols <- sapply(numeric_data, function(x) sd(x, na.rm = TRUE) == 0)

# Only keeps the columns in numeric_data that do not have zero variance.

numeric_data_filtered <- numeric_data[, !zero_var_cols]

# Create a correlation matrix of all the relevant numeric variables

cor_matrix <- cor(numeric_data, use = "pairwise.complete.obs")
print(cor_matrix)
```

```
##                      TotalMinutesAsleep.x TimeInBedAwake    WeightKg
## TotalMinutesAsleep.x          1.000000000   -0.001334242  0.03572671
## TimeInBedAwake               -0.001334242    1.000000000  0.10182853
## WeightKg                      0.035726715    0.101828533  1.00000000
## BMI                          -0.064531995    0.073961387  0.75204413
## TotalSteps                   -0.187220464    0.026720704  0.14066142
## VeryActiveMinutes            -0.091108385   -0.077774997  0.30771486
## FairlyActiveMinutes          -0.244267455    0.318151203 -0.31501245
## LightlyActiveMinutes          0.033251329   -0.147616598 -0.26323294
## SedentaryMinutes             -0.599613188   -0.165831732  0.50157699
## Calories                     -0.027930646   -0.288504744  0.58564920
## TotalActiveMinutes           -0.063660691   -0.092633259 -0.16042018
## SedentaryAwakeTime           -0.887624315   -0.027468526  0.30527963
## AvgHeartRate                 -0.258957552   -0.239026113  0.21448447
##                              BMI TotalSteps VeryActiveMinutes
## TotalMinutesAsleep.x -0.06453199 -0.1872205       -0.09110839
## TimeInBedAwake        0.07396139  0.0267207       -0.07777500
## WeightKg              0.75204413  0.1406614        0.30771486
## BMI                   1.00000000 -0.2475269       -0.05806727
## TotalSteps           -0.24752692  1.0000000        0.67729121
## VeryActiveMinutes    -0.05806727  0.6772912        1.00000000
## FairlyActiveMinutes  -0.23645806  0.3543171        0.22941847
## LightlyActiveMinutes -0.37426708  0.5945910        0.09687130
## SedentaryMinutes      0.25401247 -0.3564955       -0.18339667
## Calories              0.09891451  0.5828115        0.59486362
## TotalActiveMinutes   -0.38932775  0.7702760        0.38377977
## SedentaryAwakeTime    0.35637940 -0.1503423       -0.03634877
## AvgHeartRate          0.08401093  0.2332076        0.11035109
##                      FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes
## TotalMinutesAsleep.x        -0.244267455           0.03325133       -0.5996132
## TimeInBedAwake               0.318151203          -0.14761660       -0.1658317
## WeightKg                    -0.315012447          -0.26323294        0.5015770
## BMI                         -0.236458055          -0.37426708        0.2540125
## TotalSteps                   0.354317083           0.59459095       -0.3564955
## VeryActiveMinutes            0.229418469           0.09687130       -0.1833967
## FairlyActiveMinutes          1.000000000           0.11209072       -0.2054211
## LightlyActiveMinutes         0.112090722           1.00000000       -0.4827047
## SedentaryMinutes            -0.205421091          -0.48270471        1.0000000
## Calories                     0.304282496           0.29223589       -0.1304318
## TotalActiveMinutes           0.364838432           0.93407700       -0.5165498
## SedentaryAwakeTime          -0.003916876          -0.30394856        0.9262972
## AvgHeartRate                 0.138547592           0.18800374        0.4320997
##                         Calories TotalActiveMinutes SedentaryAwakeTime
## TotalMinutesAsleep.x -0.02793065        -0.06366069       -0.887624315
## TimeInBedAwake       -0.28850474        -0.09263326       -0.027468526
## WeightKg              0.58564920        -0.16042018        0.305279630
## BMI                   0.09891451        -0.38932775        0.356379397
## TotalSteps            0.58281154         0.77027597       -0.150342287
## VeryActiveMinutes     0.59486362         0.38377977       -0.036348769
## FairlyActiveMinutes   0.30428250         0.36483843       -0.003916876
## LightlyActiveMinutes  0.29223589         0.93407700       -0.303948561
## SedentaryMinutes     -0.13043182        -0.51654982        0.926297202
## Calories              1.00000000         0.47120072       -0.081361771
## TotalActiveMinutes    0.47120072         1.00000000       -0.299763606
## SedentaryAwakeTime   -0.08136177        -0.29976361        1.000000000
## AvgHeartRate          0.18903271         0.22900473         0.194388992
##                      AvgHeartRate
## TotalMinutesAsleep.x  -0.25895755
## TimeInBedAwake        -0.23902611
## WeightKg               0.21448447
## BMI                    0.08401093
## TotalSteps             0.23320757
## VeryActiveMinutes      0.11035109
## FairlyActiveMinutes    0.13854759
## LightlyActiveMinutes   0.18800374
## SedentaryMinutes       0.43209974
## Calories               0.18903271
## TotalActiveMinutes     0.22900473
## SedentaryAwakeTime     0.19438899
## AvgHeartRate           1.00000000
```

```
# Create a plot of the correlation matrix for clearer results.

corrplot(cor_matrix,
         method = "color",       # Use colored tiles instead of numbers or circles
         type = "upper",         # Show only the upper triangle (to avoid repetition)
         tl.cex = 0.8,           # Text label size for variable names
         number.cex = 0.5,       # Size of the numbers (correlation values)
         addCoef.col = "black",  # Color of coefficients
         tl.srt      = 55,       #
         tl.col = "darkred")     # rotate labels 45 degrees
```

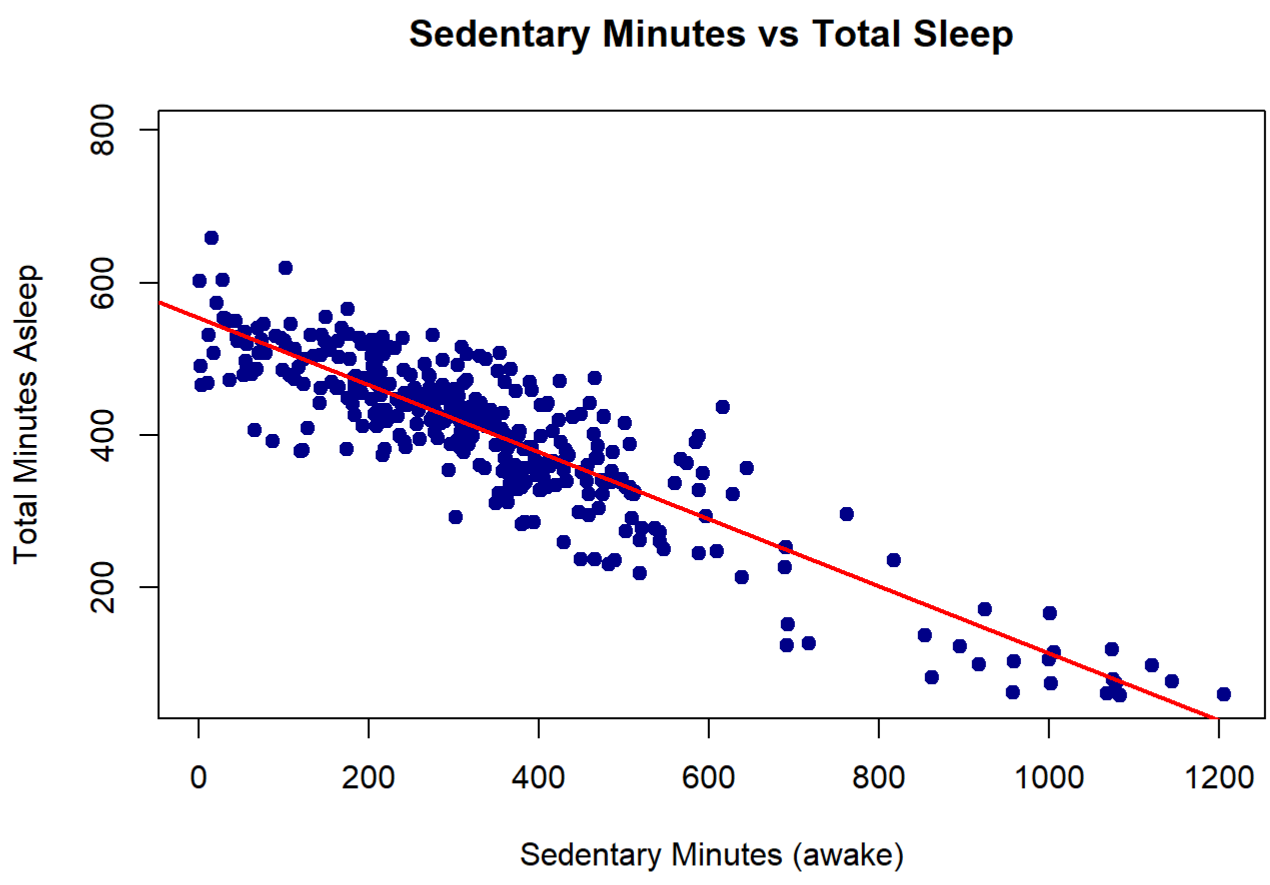


1- Sedentary time correlate with total sleeping time (without accounting for sedentary time spend sleeping).

```
cor.test(Activity$SedentaryAwakeTime, Activity$TotalMinutesAsleep, use = "complete.obs")
```

```
##
##  Pearson's product-moment correlation
##
## data:  Activity$SedentaryAwakeTime and Activity$TotalMinutesAsleep
## t = -36.97, df = 364, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.9084037 -0.8649250
## sample estimates:
##        cor
## -0.8886447
```

```
# r(342) = -0.89, p < .001 (Very strong negative correlation/Very significant).

plot(Activity$SedentaryAwakeTime, Activity$TotalMinutesAsleep,
     main = "Sedentary Minutes vs Total Sleep",
     xlab = "Sedentary Minutes (awake)",
     ylab = "Total Minutes Asleep",
     pch = 19, col = "darkblue")
abline(lm(TotalMinutesAsleep ~ SedentaryAwakeTime, data = Activity), col = "red", lwd = 2) #
adding a trend line (linear regression)
```



```
## As sedentary minutes increase, sleep minutes tend to decrease and vice versa.
```

2- Heart rate correlate with sleeping time

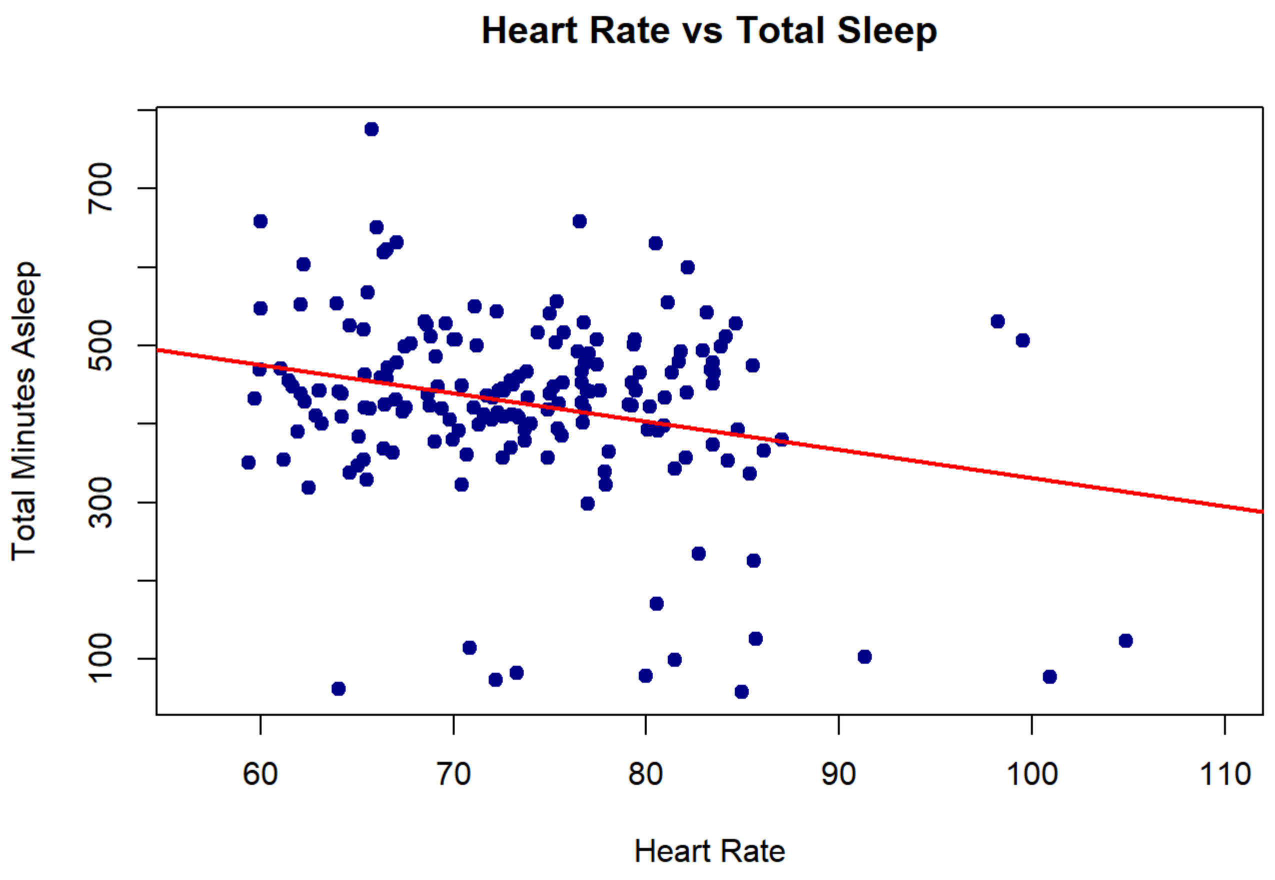
```
Heartrate_daily <- Heartrate %>%
  group_by(Id, Date) %>%
  summarize(AvgHeartRate = mean(Value, na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'Id'. You can override using the `.groups`
## argument.
```

```
# Aggregate the Heart Rate data by computing the daily average, in order to reduce multiple v
alues per day and match them with the corresponding single daily Sleep record.

HeartbeatSleep <- merge(Heartrate_daily, Activity, by= c("Id", "Date"))

cor.test(HeartbeatSleep$AvgHeartRate, HeartbeatSleep$TotalMinutesAsleep)
```

```
##
##  Pearson's product-moment correlation
##
## data:  HeartbeatSleep$AvgHeartRate and HeartbeatSleep$TotalMinutesAsleep
## t = -3.5283, df = 179, p-value = 0.0005314
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.3864854 -0.1133612
## sample estimates:
##        cor
## -0.2550026
```

```
# r(179)= -0.26, p<.001 (Weak negative correlation, very significant).


plot(HeartbeatSleep$AvgHeartRate, HeartbeatSleep$TotalMinutesAsleep,
     main = "Heart Rate vs Total Sleep",
     xlab = "Heart Rate",
     ylab = "Total Minutes Asleep",
     pch = 19, col = "darkblue")
abline(lm(TotalMinutesAsleep ~ AvgHeartRate, data = HeartbeatSleep), col = "red", lwd = 2) #
adding a trend line (linear regression)
```



```
# The more people spend time sleeping, the lower is their heart rate and vice versa.
```

3- Heart Rate correlates with Sedentary time

```
HeartrateActivity <- merge(Heartrate_daily, Activity, by= c("Id", "Date")) # using awake sede
ntary time rather than sedentary time (including sleeping time) as Sleep naturally lowers hea
rt rate (due to parasympathetic activity)

cor.test(HeartrateActivity$AvgHeartRate, HeartrateActivity$SedentaryAwakeTime)
```

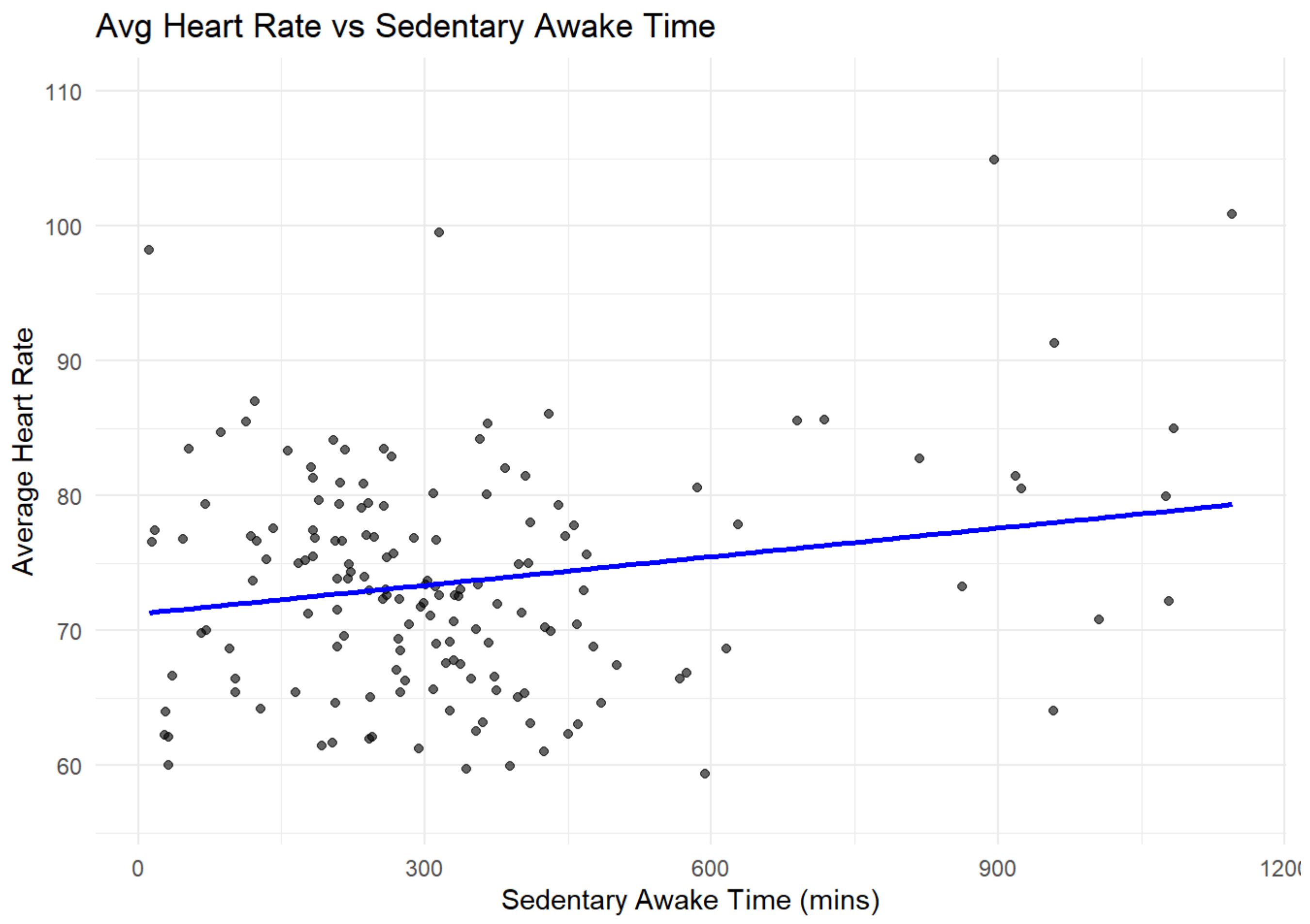
```
## 
##  Pearson's product-moment correlation
## 
## data:  HeartrateActivity$AvgHeartRate and HeartrateActivity$SedentaryAwakeTime
## t = 2.5124, df = 156, p-value = 0.01301
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.0423680 0.3427891
## sample estimates:
##       cor 
## 0.1972037
```

```
# r(156)= 0.20, p<.05 (Weak positive correlation, statistically significant)

ggplot(HeartrateActivity, aes(x = SedentaryAwakeTime, y = AvgHeartRate)) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = FALSE, color = "blue") +
  labs(title = "Avg Heart Rate vs Sedentary Awake Time",
       x = "Sedentary Awake Time (mins)",
       y = "Average Heart Rate") +
  theme_minimal()
```

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

```
## Warning: Removed 311 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 311 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



```
# People with more sedentary time tend to have slightly higher heart rates and vice versa.
```

4- Total Active Minutes and Steps

```
cor.test(Activity$TotalActiveMinutes, Activity$TotalSteps)
```

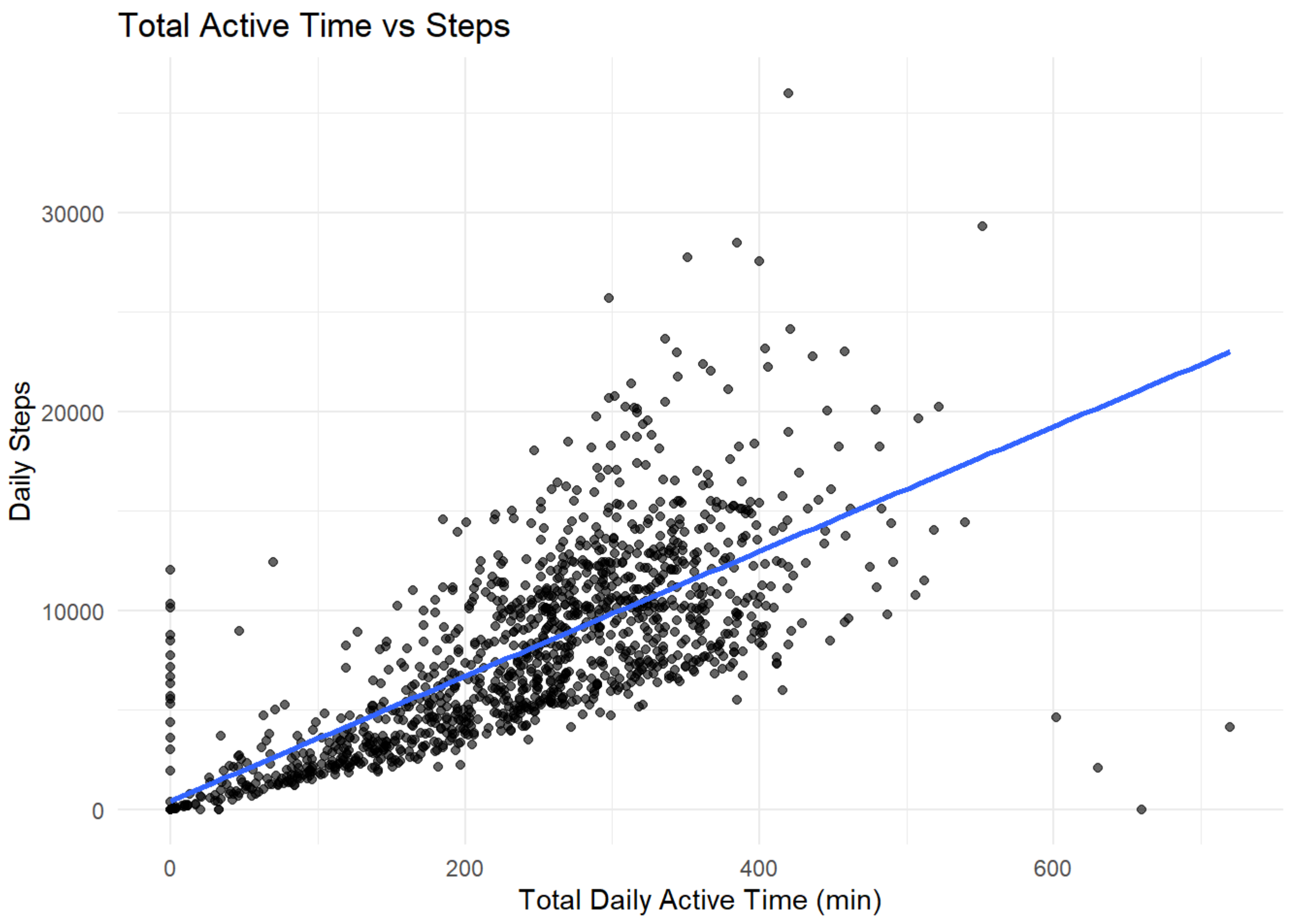
```
##
##  Pearson's product-moment correlation
##
## data:  Activity$TotalActiveMinutes and Activity$TotalSteps
## t = 44.682, df = 1371, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.7475305 0.7906786
## sample estimates:
##       cor
## 0.7699834
```

```r
# r(1371)= 0.77, p<.001 (Very strong positive correlation, very significant)


ggplot(Activity, aes(x= TotalActiveMinutes, y= Activity$TotalSteps))+
  geom_point(alpha= 0.6)+
  geom_smooth(method = "lm", se= FALSE, Color="blue")+
  labs(title= "Total Active Time vs Steps",
       x= "Total Daily Active Time (min)",
       y= "Daily Steps")+
  theme_minimal()
```

```
## Warning in geom_smooth(method = "lm", se = FALSE, Color = "blue"): Ignoring
## unknown parameters: `Color`
```

```
## Warning: Use of `Activity$TotalSteps` is discouraged.
## ℹ Use `TotalSteps` instead.
## Use of `Activity$TotalSteps` is discouraged.
## ℹ Use `TotalSteps` instead.
```

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



```
# The correlation between Total Active Minutes and Total Steps is very strong (r = 0.77, p <
.001), indicating a significant and positive linear relationship. This suggests that individu
als who spend more time being active tend to take more steps throughout the day. The strong c
orrelation also implies that a large portion of daily physical activity is likely made up of
walking or stepping-based movement, reinforcing the role of step count as a key component of
overall activity levels.
```

5- Calories and Very Active minutes

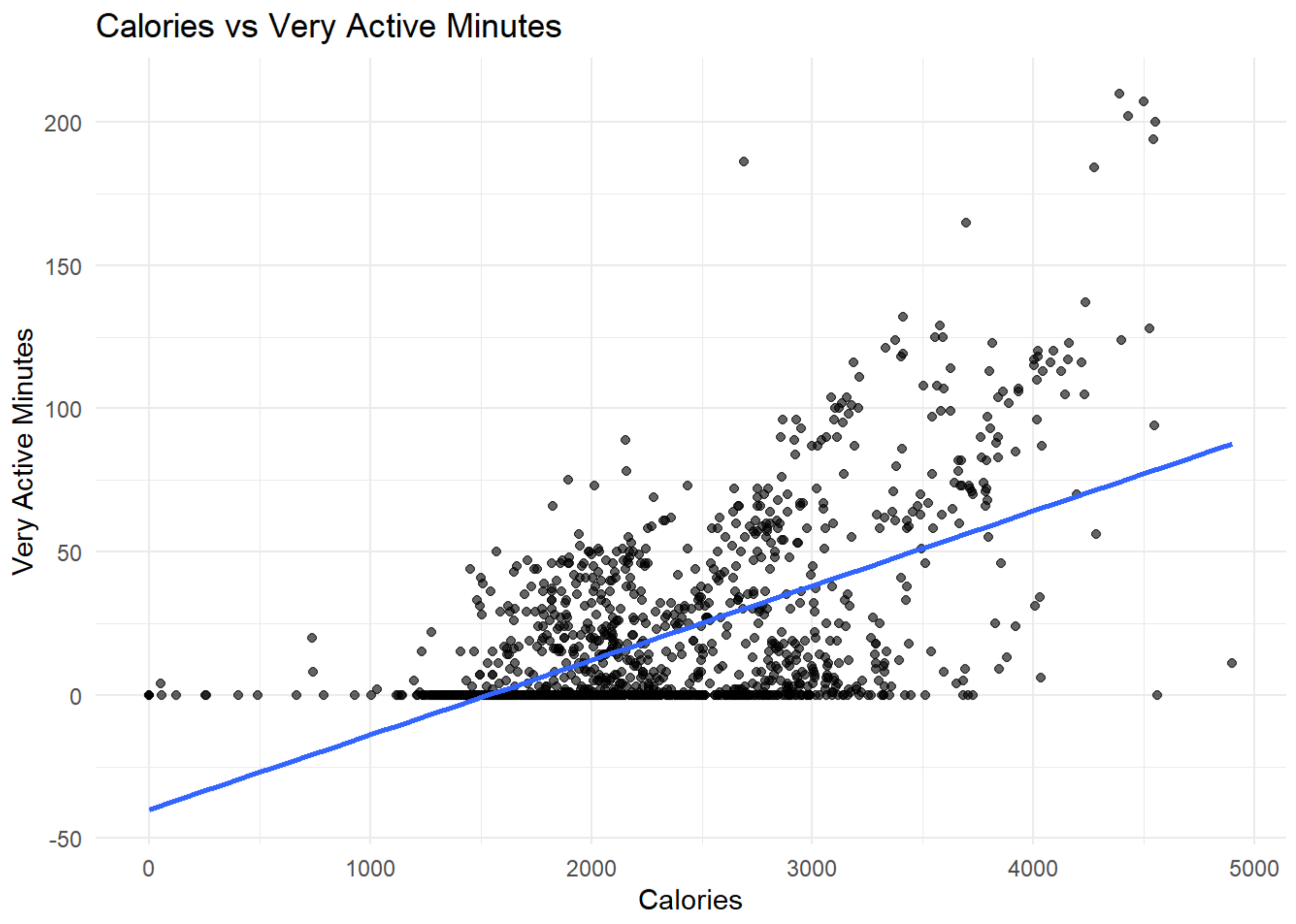
```
cor.test(Activity$Calories, Activity$VeryActiveMinutes)
```

```
##
##  Pearson's product-moment correlation
##
## data:  Activity$Calories and Activity$VeryActiveMinutes
## t = 27.328, df = 1371, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.5584704 0.6270336
## sample estimates:
##      cor
## 0.593829
```

```
# r(1371) =0.59p<.001 (positive correlation, Very significant)

ggplot(Activity, aes(x=Calories, y=VeryActiveMinutes))+
  geom_point(alpha=0.6)+ #60% opaque
  geom_smooth(method = "lm", se= FALSE, Color="blue")+ # linear model, False = hides shaded c
onfidence interval
  labs(title= "Calories vs Very Active Minutes",
       x= "Calories",
       y="Very Active Minutes")+
  theme_minimal()
```

```
## Warning in geom_smooth(method = "lm", se = FALSE, Color = "blue"): Ignoring
## unknown parameters: `Color`
```

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



```
# As number of Very Active minutes increase, burnt Calories increases
```

6- Calories and Steps

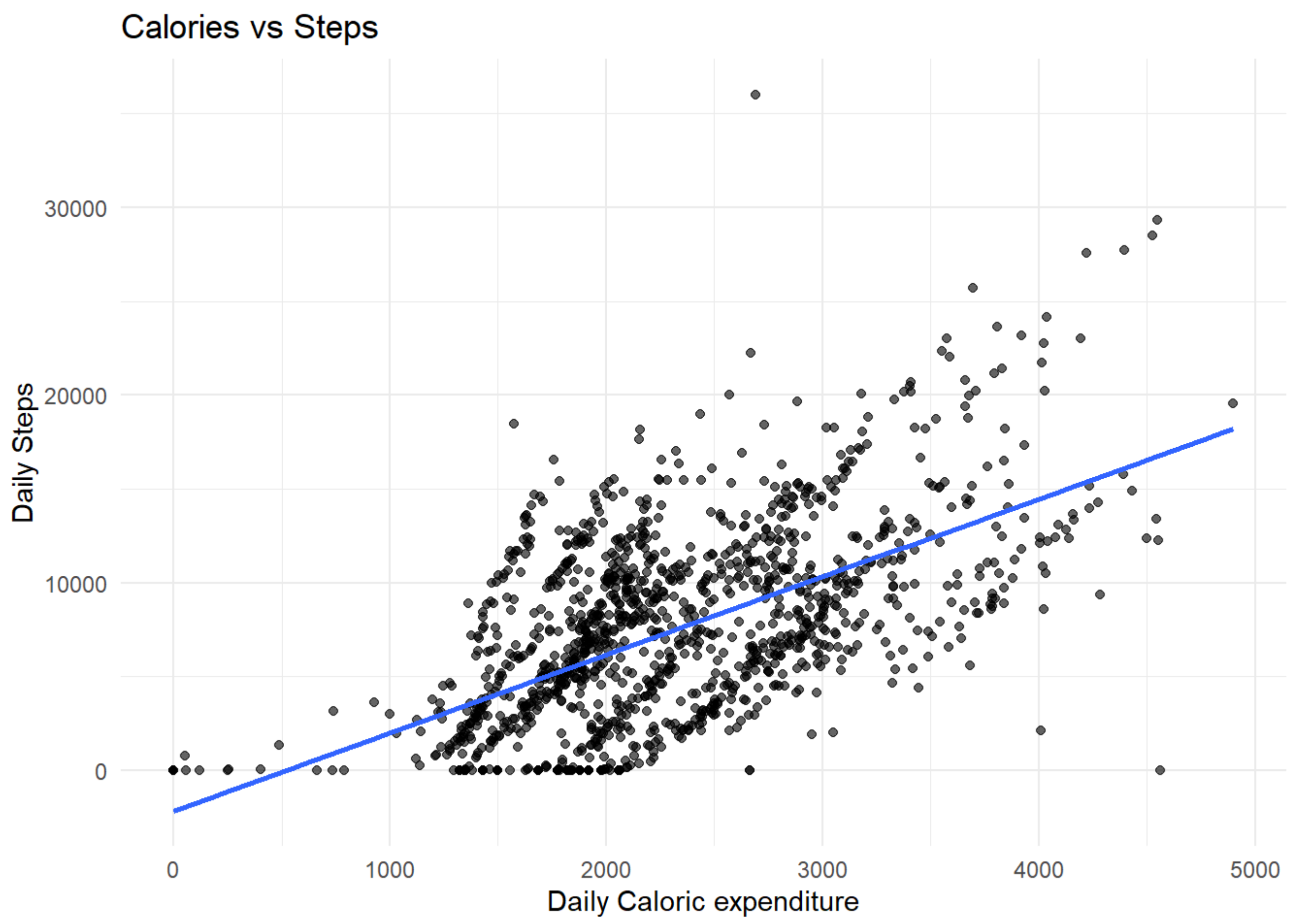
```
cor.test(Activity$Calories, Activity$TotalSteps)
```

```
##
##  Pearson's product-moment correlation
##
## data:  Activity$Calories and Activity$TotalSteps
## t = 26.368, df = 1371, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.5438633 0.6141333
## sample estimates:
##       cor
## 0.5800765
```

```
# r(1371)=0.58, p<.001 (positive correlation, very significant)

ggplot(Activity, aes(x=Calories, y=TotalSteps))+
  geom_point(alpha=0.6)+
  geom_smooth(method = "lm", se= FALSE, Color="blue")+
  labs(title = "Calories vs Steps",
       x="Daily Caloric expenditure",
       y="Daily Steps")+
  theme_minimal()
```

```
## Warning in geom_smooth(method = "lm", se = FALSE, Color = "blue"): Ignoring
## unknown parameters: `Color`
```

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



```
# There is a positive relationship between daily step count and calorie burn, indicating that
individuals who walk more steps tend to expend more calories throughout the day.
```

7- Time in Bed (Awake) and Fairly Active minutes

```
cor.test(Allmerged$TimeInBedAwake, Allmerged$FairlyActiveMinutes)
```

```
##
##  Pearson's product-moment correlation
##
## data:  Allmerged$TimeInBedAwake and Allmerged$FairlyActiveMinutes
## t = 6.8117, df = 412, p-value = 3.43e-11
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2287887 0.4021966
## sample estimates:
##       cor
## 0.3181512
```

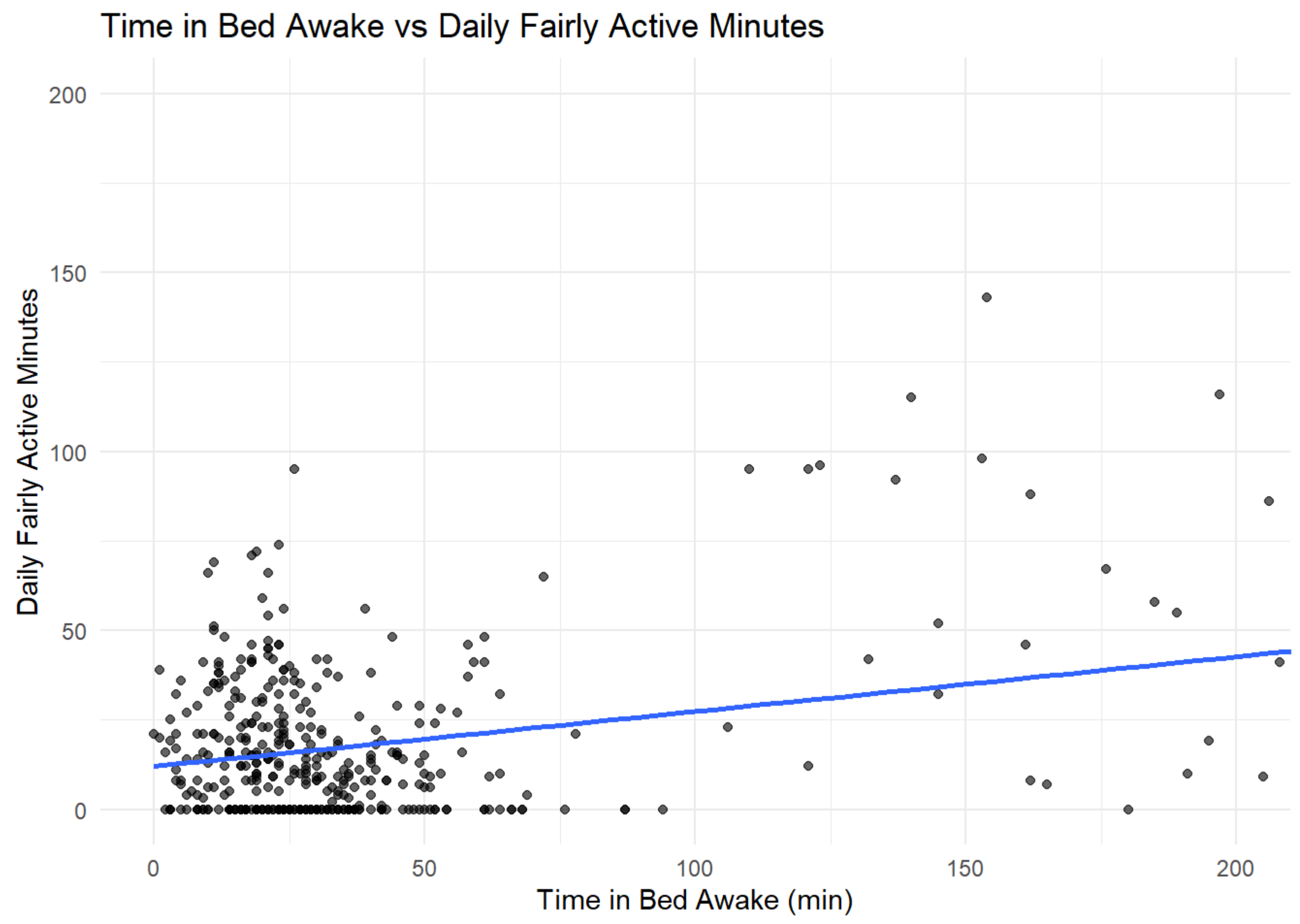
```
# r(412)=0.32, p<.001 (weak positive correlation, very significant)



ggplot(Allmerged, aes(x=TimeInBedAwake, y=FairlyActiveMinutes))+
  geom_point(alpha=0.6)+
  geom_smooth(method="lm", se=FALSE, Color="blue")+
  labs(title="Time in Bed Awake vs Daily Fairly Active Minutes",
       x="Time in Bed Awake (min)",
       y="Daily Fairly Active Minutes")+
  coord_cartesian(xlim = c(0, 200), ylim = c(0, 200)) +  # zoom to focus area without droppin
g data
  theme_minimal()
```

```
## Warning in geom_smooth(method = "lm", se = FALSE, Color = "blue"): Ignoring
## unknown parameters: `Color`
```

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

```
## Warning: Removed 964 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 964 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



```
#The more time Daily fairly active time is spent the more time in bed is spent before and aft
er sleeping and vice versa. Because correlation does not imply causation, this association hi
ghlights a connection but doesn’t clarify the direction of influence. Those results could ind
icate that spending time in bed helps relaxation and increases motivation for the rest of the
day resulting in higher fairly active time but it could also indicate that longer fairly acti
ve time leads to the need of more time to relax once in bed.
```

Summary of Analysis

Key findings from the data analysis include:

Activity Trends: The average daily step count was 7,281—well below the recommended 10,000. Users were more active on Saturdays and least active on Fridays and Sundays. Weekends showed slightly more active minutes and lower sedentary behavior.

Sleep Patterns: Users sleep longer on weekends, especially on Sundays. A significant portion of time in bed is spent awake, particularly on weeknights.

Heart Rate Insights: Slightly elevated average heart rate on weekends. Weak negative correlation between heart rate and sleep duration.

Weight and BMI: Slight increase in weight and BMI over weekends, peaking on Sundays and Wednesdays. Correlation Highlights: Strong positive correlation between total active minutes and step count (r = 0.77). Moderate positive correlation between very active minutes and calories burned. Strong negative correlation between sedentary awake time and total sleep (r = -0.89). Slight positive correlation between sedentary awake time and heart rate.

Key Insight Summaries: Less activity correlates with reduced sleep and slightly higher heart rate. Most users do not meet daily activity guidelines. User behaviors differ significantly between weekdays and weekends, affecting sleep, heart rate, and calorie burn.

Based on the analysis, the following strategic recommendations are proposed for Bellabeat:

1. Introduce Personalized Activity Nudges

Insight: Most users do not meet the 10,000-step daily recommendation; activity is lowest on weekdays,especially Fridays. Action: Use app notifications to encourage short bursts of movement during sedentary periods, especially during workdays. Example: “Take 1,000 steps by 3 PM to hit your daily goal!”

2. Emphasize Weekend Wellness Features

Insight: Users sleep more and are slightly more active on weekends. Action: Launch weekend wellness campaigns featuring mindfulness, recovery, hydration tracking, and sleep rituals. “Recharge Sundays”:Guided breathing + hydration reminders + sleep prep content.

3. Create Smart Sleep Coaching

Insight: There’s a strong negative correlation between sedentary time and sleep; many users spend time in bed awake. Action: Offer in-app sleep coaching to help users improve sleep efficiency (not just duration), including: Pre-bed routines: Light/stretch reminders before sleep. Content to reduce “awake in bed” time (e.g., sleep stories, relaxation sounds)

4. Launch Heart Health Awareness Tools

Insight: Higher sedentary time is weakly correlated with higher heart rate. Action: Introduce heart rate monitoring insights that detect patterns and offer tips (e.g., breathing exercises, movement alerts).“We noticed your heart rate is elevated—want to take a 3-minute breathing break?”

5. Add Adaptive Daily Goals

Insight: Activity and sleep levels vary by day of the week. Action: Use machine learning to adapt daily step or sleep goals based on user trends (e.g., higher weekend activity). Adjusted expectations may improve user satisfaction and consistency.

6. Implement “Lifestyle Snapshot” Dashboards

Insight: Correlation analysis showed meaningful connections between activity, heart rate, sleep, and calories. Action: Provide users with a weekly health summary combining: Steps, Heart rate trends, Sleep efficiency, Calories burned. Example: “This week you were most active on Saturday. Sleep improved on days with less sedentary time.”

7. Gamify Movement & Sleep

Insight: Users fall short on physical activity and may benefit from motivation. Action: Add achievements, badges, and streaks for: Meeting step goals, Improving sleep efficiency, Lowering resting heart rate over time.

Area for future research

1. Audience Segmentation

Further research could explore how different age groups of female users engage with wearable technology. Specifically: Do younger women (ages 20–35) prioritize fitness tracking, step goals, and the visual/aesthetic appeal of devices? Do older women (ages 35+) place more value on features that support wellness, heart health, sleep monitoring, and stress management?

2. Competitive Differentiation

Further exploration could assess whether Bellabeat can create a stronger market position by focusing on holistic wellness rather than traditional fitness tracking alone. Specifically: Do users respond more positively to devices that support both physical and mental wellbeing, such as stress reduction, sleep quality, mindfulness, and emotional balance? Would framing Bellabeat as a lifestyle partner—rather than a step counter—better align with the needs and expectations of its core audience?

3. Female-Specific Health Features

Further development could explore how Bellabeat can better support women’s unique health needs by integrating features tailored to different life stages. Specifically: Do users benefit from tracking tools related to menstrual cycles, fertility windows, and hormonal fluctuations? Could women entering perimenopause and menopause find value in tools focused on stress management, sleep support, and holistic wellness guidance?

Understanding these preferences could help Bellabeat tailor its product features and marketing messages to better meet the distinct needs of these segments.