Case_Study_Bellabeat_Report

Daniel Marin

2025-07-21

Business Task

The marketing analytics team at Bellabeat, a high-tech company focused on women's health, was appointed by Urška Sršen to investigate how users interact with the company's products. Sršen firmly believes that the data generated by these devices can uncover valuable insights to strategically grow the business.

As a junior analyst on the team, I have been assigned to analyze one of Bellabeat's products in order to understand how users engage with it. The goal is to apply these findings to the brand's current customer profile and provide the marketing team with a strong data foundation to support future strategic decisions.

This analysis aims to answer three key questions:

- What trends exist in the usage of smart devices?
- How could these trends be applied to Bellabeat users?
- How might these trends influence the company's marketing strategy?

2. Prepare: Data Description

For this case study, a public dataset available on Kaggle, published by user Arash Nic, was used. This dataset is licensed under CC0 (public domain), allowing free use for educational and analytical purposes.

The downloaded file was a .zip containing multiple .csv files organized into folders by export date. Two main folders were identified: mturkfitbit_export_3.12.16 and mturkfitbit_export_4.12.16. Both contain similar files with continuous time coverage. For the analysis, both folders were combined after confirming that the data from 4.12.16 continues where the 3.12.16 data ends.

The files are in wide format, where each row represents an observation per user and date. The data was stored locally and loaded into R for analysis.

During the initial exploration, it was noted that not all users appear in every file, likely because each user enabled different Fitbit features (e.g., sleep tracking, heart rate, weight, etc.). This finding was taken into account to avoid incorrect joins or data loss.

Based on the ROCCC criteria, the data was assessed as:

Reliable: collected directly from real devices used by 30 individuals.

Original: not modified by third parties.

Comprehensive: includes minute-, hour-, and day-level data.

Current: although from 2016, it remains useful for exploring general smart device usage habits.

Cited: properly referenced on Kaggle.

No additional data sources will be used at this stage, though they may be considered later to enhance the analysis.

3. Process: Cleaning and Transformation

3.1 Loading Packages and Files

```
library(tidyverse)
library(lubridate)
library(dplyr)
library(broom)
library(tidyr)
```

```
# Load main files (from both folders)
df_fitbit_3.12.16 <- read_csv("mturkfitbit_export_3.12.16-4.11.16/Fitabase Data 3.12.16-4.11.16/dailyAc
```

```
## Rows: 457 Columns: 15
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): ActivityDate
## dbl (14): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesDi...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_fitbit_4.12.16 <- read_csv("mturkfitbit_export_4.12.16-5.12.16/Fitabase Data 4.12.16-5.12.16/dailyAc
```

```
## Rows: 940 Columns: 15
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): ActivityDate
## dbl (14): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesDi...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_SleepMinute_3.12 <- read_csv("mturkfitbit_export_3.12.16-4.11.16/Fitabase Data 3.12.16-4.11.16/minut
```

```
## Rows: 198559 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): date
## dbl (3): Id, value, logId
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_SleepDay_4.12 <- read_csv("mturkfitbit_export_4.12.16-5.12.16/Fitabase Data 4.12.16-5.12.16/sleepDay
```

```
## Rows: 413 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
```

```
## chr (1): SleepDay
## dbl (4): Id, TotalSleepRecords, TotalMinutesAsleep, TotalTimeInBed
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_minuteMETs_3.12 <- read_csv("mturkfitbit_export_3.12.16-4.11.16/Fitabase Data 3.12.16-4.11.16/minuteM
```

```
## Rows: 1445040 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ActivityMinute
## dbl (2): Id, METs
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_minuteMETs_4.12 <- read_csv("mturkfitbit_export_4.12.16-5.12.16/Fitabase Data 4.12.16-5.12.16/minuteM
```

```
## Rows: 1325580 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ActivityMinute
## dbl (2): Id, METs
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_weightLog_3.12 <- read_csv("mturkfitbit_export_3.12.16-4.11.16/Fitabase Data 3.12.16-4.11.16/weightLo
```

```
## Rows: 33 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Date
## dbl (6): Id, WeightKg, WeightPounds, Fat, BMI, LogId
## lgl (1): IsManualReport
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_weightLog_4.12 <- read_csv("mturkfitbit_export_4.12.16-5.12.16/Fitabase Data 4.12.16-5.12.16/weightLo
```

```
## Rows: 67 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Date
## dbl (6): Id, WeightKg, WeightPounds, Fat, BMI, LogId
## lgl (1): IsManualReport
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

3.2 Inspecting the Data

We inspected the data that appeared most relevant to answering the key questions of the project. We noticed that in the 3.12 folder, the sleep data is recorded by minute, whereas in the 4.12 folder, it is summarized by day. To work with both datasets consistently, we will first standardize them to a daily format. Then, we will merge the Daily Activity files from both folders and finally consolidate the Sleep and Daily Activity data into a unified dataset.

```
#Data inspect

head(df_fitbit_3.12.16)
```

```
## # A tibble: 6 x 15
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
##        <dbl> <chr>              <dbl>         <dbl>           <dbl>
## 1 1503960366 3/25/2016         11004          7.11            7.11
## 2 1503960366 3/26/2016         17609         11.6            11.6
## 3 1503960366 3/27/2016         12736          8.53            8.53
## 4 1503960366 3/28/2016         13231          8.93            8.93
## 5 1503960366 3/29/2016         12041          7.85            7.85
## 6 1503960366 3/30/2016         10970          7.16            7.16
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

```
head(df_fitbit_4.12.16)
```

```
## # A tibble: 6 x 15
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
##        <dbl> <chr>              <dbl>         <dbl>           <dbl>
## 1 1503960366 4/12/2016         13162          8.5             8.5
## 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
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

```
head(df_SleepMinute_3.12)
```

```
## # A tibble: 6 x 4
##           Id date                 value       logId
##        <dbl> <chr>                <dbl>       <dbl>
## 1 1503960366 3/13/2016 2:39:30 AM     1 11114919637
## 2 1503960366 3/13/2016 2:40:30 AM     1 11114919637
## 3 1503960366 3/13/2016 2:41:30 AM     1 11114919637
```

```
## 4 1503960366 3/13/2016 2:42:30 AM     1 11114919637
## 5 1503960366 3/13/2016 2:43:30 AM     1 11114919637
## 6 1503960366 3/13/2016 2:44:30 AM     1 11114919637
```

```
head(df_SleepDay_4.12)
```

```
## # A tibble: 6 x 5
##           Id SleepDay        TotalSleepRecords TotalMinutesAsleep TotalTimeInBed
##        <dbl> <chr>                       <dbl>              <dbl>          <dbl>
## 1 1503960366 4/12/2016 12:0~                 1                327            346
## 2 1503960366 4/13/2016 12:0~                 2                384            407
## 3 1503960366 4/15/2016 12:0~                 1                412            442
## 4 1503960366 4/16/2016 12:0~                 2                340            367
## 5 1503960366 4/17/2016 12:0~                 1                700            712
## 6 1503960366 4/19/2016 12:0~                 1                304            320
```

```
head(df_minuteMETs_3.12)
```

```
## # A tibble: 6 x 3
##           Id ActivityMinute         METs
##        <dbl> <chr>                 <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM    10
## 2 1503960366 3/12/2016 12:01:00 AM    10
## 3 1503960366 3/12/2016 12:02:00 AM    10
## 4 1503960366 3/12/2016 12:03:00 AM    10
## 5 1503960366 3/12/2016 12:04:00 AM    10
## 6 1503960366 3/12/2016 12:05:00 AM    10
```

```
head(df_minuteMETs_4.12)
```

```
## # A tibble: 6 x 3
##           Id ActivityMinute         METs
##        <dbl> <chr>                 <dbl>
## 1 1503960366 4/12/2016 12:00:00 AM    10
## 2 1503960366 4/12/2016 12:01:00 AM    10
## 3 1503960366 4/12/2016 12:02:00 AM    10
## 4 1503960366 4/12/2016 12:03:00 AM    10
## 5 1503960366 4/12/2016 12:04:00 AM    10
## 6 1503960366 4/12/2016 12:05:00 AM    12
```

```
head(df_weightLog_3.12)
```

```
## # A tibble: 6 x 8
##           Id Date       WeightKg WeightPounds   Fat   BMI IsManualReport   LogId
##        <dbl> <chr>         <dbl>        <dbl> <dbl> <dbl> <lgl>            <dbl>
## 1 1503960366 4/5/2016 ~     53.3         118.    22  23.0 TRUE           1.46e12
## 2 1927972279 4/10/2016~    130.          286.    NA  46.2 FALSE          1.46e12
## 3 2347167796 4/3/2016 ~     63.4         140.    10  24.8 TRUE           1.46e12
## 4 2873212765 4/6/2016 ~     56.7         125.    NA  21.5 TRUE           1.46e12
## 5 2873212765 4/7/2016 ~     57.2         126.    NA  21.6 TRUE           1.46e12
## 6 2891001357 4/5/2016 ~     88.4         195.    NA  25.0 TRUE           1.46e12
```

```r
head(df_weightLog_4.12)
```

```
## # A tibble: 6 x 8
##           Id Date       WeightKg WeightPounds   Fat   BMI IsManualReport   LogId
##        <dbl> <chr>         <dbl>        <dbl> <dbl> <dbl> <lgl>            <dbl>
## 1 1503960366 5/2/2016 ~     52.6         116.    22  22.6 TRUE           1.46e12
## 2 1503960366 5/3/2016 ~     52.6         116.    NA  22.6 TRUE           1.46e12
## 3 1927972279 4/13/2016~    134.          294.    NA  47.5 FALSE          1.46e12
## 4 2873212765 4/21/2016~     56.7         125.    NA  21.5 TRUE           1.46e12
## 5 2873212765 5/12/2016~     57.3         126.    NA  21.7 TRUE           1.46e12
## 6 4319703577 4/17/2016~     72.4         160.    25  27.5 TRUE           1.46e12
```

3.3 Data Standardization

The main files were inspected using head() to get a general overview of their structure and contents.

Then, date columns were standardized across all relevant tables to ensure consistent formatting. Some columns contained datetime values, which were converted to simple Date format as needed for the analysis.

```r
# Date Standardize

df_fitbit_3.12.16 <- df_fitbit_3.12.16 %>%
  mutate(ActivityDate = mdy(ActivityDate))

df_fitbit_4.12.16 <- df_fitbit_4.12.16 %>%
  mutate(ActivityDate = mdy(ActivityDate))

df_SleepDay_4.12 <- df_SleepDay_4.12 %>%
  mutate(SleepDay = mdy_hms(SleepDay)) %>%
  mutate(SleepDay = as_date(SleepDay))

df_SleepMinute_3.12 <- df_SleepMinute_3.12 %>%
  mutate(date = mdy_hms(date)) %>%
  mutate(SleepDay = as_date(date))

df_minuteMETs_3.12 <- df_minuteMETs_3.12 %>%
  mutate(ActivityMinute = mdy_hms(ActivityMinute)) %>%
  mutate(ActivityMinute = as_date(ActivityMinute)) %>%
  rename(ActivityDate = ActivityMinute)

df_minuteMETs_4.12 <- df_minuteMETs_4.12 %>%
  mutate(ActivityMinute = mdy_hms(ActivityMinute)) %>%
  mutate(ActivityMinute = as_date(ActivityMinute)) %>%
  rename(ActivityDate = ActivityMinute)

df_weightLog_3.12 <- df_weightLog_3.12 %>%
  mutate(Date = mdy_hms(Date)) %>%
  mutate(Date = as_date(Date))

df_weightLog_4.12 <- df_weightLog_4.12 %>%
  mutate(Date = mdy_hms(Date)) %>%
  mutate(Date = as_date(Date))
```

3.4 Data Preparation

Sleep Data Minute-level sleep data was aggregated to a daily format. For each user and day, the following metrics were calculated:

Total minutes asleep (value == 1)

Total time in bed (number of records per day)

One sleep record per user per day (TotalSleepRecords = 1)

The sleep data from both folders was merged, and overlapping dates were summed to avoid information loss.

```
# We group by user and date to calculate:
# - Total minutes slept (value == 1)
# - Total minutes in bed (all records)
# - Number of records (simulating TotalSleepRecords = 1 per day)

df_SleepDay_3.12 <- df_SleepMinute_3.12 %>%
  group_by(Id, SleepDay) %>%
  summarise(
    TotalSleepRecords = 1,
    TotalMinutesAsleep = sum(value == 1),
    TotalTimeInBed = n(),
    .groups = "drop"
  )

# Join both tables
DailySleep <- bind_rows(df_SleepDay_3.12, df_SleepDay_4.12)

# Add the data from overlapping days to avoid data loss

DailySleep <- DailySleep %>%
  group_by(Id, SleepDay) %>%
  summarise(
    TotalSleepRecords = sum(TotalSleepRecords, na.rm = TRUE),
    TotalMinutesAsleep = sum(TotalMinutesAsleep, na.rm = TRUE),
    TotalTimeInBed = sum(TotalTimeInBed, na.rm = TRUE),
    .groups = "drop"
  )
```

METs Data During the inspection of the minuteMETs table, extremely high values were detected—up to 189 METs per minute, which is physiologically impossible. According to the Compendium of Physical Activities, values above 20–23 METs per minute are unrealistic.

Deduction: The values appear to be scaled by a factor of 10. Dividing them by 10 places them within plausible ranges for high-intensity physical activity. A new column METs_scaled was created, and daily averages were computed per user.

```
# Make sure you have METs already scaled (divided by 10)

df_minuteMETs_3.12 <- df_minuteMETs_3.12 %>%
  mutate(METs_scaled = METs / 10)

df_minuteMETs_4.12 <- df_minuteMETs_4.12 %>%
  mutate(METs_scaled = METs / 10)
```

```r
# Calculate daily average per user

df_minuteMETs_3.12 <- df_minuteMETs_3.12 %>%
  group_by(Id, ActivityDate) %>%
  summarise(avg_METs = mean(METs_scaled), .groups = "drop")

df_minuteMETs_4.12 <- df_minuteMETs_4.12 %>%
  group_by(Id, ActivityDate) %>%
  summarise(avg_METs = mean(METs_scaled), .groups = "drop")

# Join both tables to consolidate DailyAvgMETs

DailyAvgMETs <- bind_rows(df_minuteMETs_3.12, df_minuteMETs_4.12)

# Cleaning Overlapping Data

DailyAvgMETs <- DailyAvgMETs %>%
  group_by(Id, ActivityDate) %>%
  summarise(
    METs = sum(avg_METs, na.rm = TRUE),
    .groups = "drop"
  )
```

Weight Data The weight logs from both folders were combined. To simplify future analysis, unnecessary columns like WeightPounds, Fat, BMI, and LogId were removed.

```r
# We join both tables of Weight records

DailyWeight <- bind_rows(df_weightLog_3.12, df_weightLog_4.12)

# We eliminate columns that are not needed to join it with the table that will be created later (Full_D

DailyWeight_clean <- DailyWeight %>% select(-WeightPounds ,-Fat, -BMI, -LogId)
```

Daily Activity Data The two dailyActivity datasets were merged. Duplicate entries (by user and date) were detected and aggregated to avoid data loss while keeping daily granularity.

```r
# We have previously standardized the dates so now we will just join the tables

df_fitbit <- bind_rows(df_fitbit_3.12.16, df_fitbit_4.12.16)

# We check for duplicate data

df_fitbit %>%
  count(Id, ActivityDate) %>%
  filter(n > 1) %>%
  print(n = Inf)
```

```
## # A tibble: 24 x 3
##            Id ActivityDate     n
##         <dbl> <date>       <int>
```

```
##  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
```

```r
# And we add the duplicate data so as not to lose information.

df_fitbit <- df_fitbit %>%
  group_by(Id, ActivityDate) %>%
  summarise(
    TotalSteps = sum(TotalSteps, na.rm = TRUE),
    TotalDistance = sum(TotalDistance, na.rm = TRUE),
    TrackerDistance = sum(TrackerDistance, na.rm = TRUE),
    LoggedActivitiesDistance = sum(LoggedActivitiesDistance, na.rm = TRUE),
    VeryActiveDistance = sum(VeryActiveDistance, na.rm = TRUE),
    ModeratelyActiveDistance = sum(ModeratelyActiveDistance, na.rm = TRUE),
    LightActiveDistance = sum(LightActiveDistance, na.rm = TRUE),
    SedentaryActiveDistance = sum(SedentaryActiveDistance, na.rm = TRUE),
    VeryActiveMinutes = sum(VeryActiveMinutes, na.rm = TRUE),
    FairlyActiveMinutes = sum(FairlyActiveMinutes, na.rm = TRUE),
    LightlyActiveMinutes = sum(LightlyActiveMinutes, na.rm = TRUE),
    SedentaryMinutes = sum(SedentaryMinutes, na.rm = TRUE),
    Calories = sum(Calories, na.rm = TRUE),
    .groups = "drop"
  )
```

At this point, the following main tables are available:

df_fitbit (1373 records)

DailyAvgMETs (1935 records)

DailySleep (870 records)

DailyWeight (100 records)

The DailyAvgMETs table contains over 500 more records than the daily activity table. Upon comparing users and date ranges, it was confirmed that they match. Additionally, several users have daily average METs values of exactly 1.00000, suggesting that the function might be auto-recording even without meaningful activity.

Because of this, we decided not to merge the DailyAvgMETs table into the main dataset (Full_DailyActivity) to prevent noise in the analysis. Instead, it will be analyzed separately to better understand highly active user profiles.

```
# Users who are in METs but not in dailyActivity
unique_METs <- unique(DailyAvgMETs$Id)
unique_fitbit <- unique(df_fitbit$Id)

setdiff(unique_METs, unique_fitbit)
```

```
## numeric(0)
```

```
data.frame(
  METs = range(DailyAvgMETs$ActivityDate),
  Full_Daily = range(df_fitbit$ActivityDate)
)
```

```
##         METs Full_Daily
## 1 2016-03-12 2016-03-12
## 2 2016-05-12 2016-05-12
```

We confirmed that the data indeed corresponds to the same users and the same date range. Additionally, we observed that in the DailyAvgMETs table, several users have days with an exact average of 1.00000 METs, which suggests that the data might be automatically recorded—even without actual physical activity.

These days can be interpreted as non-usage days of the METs function. However, to avoid introducing noise into the general analysis of daily behavior, we decided not to merge this table into the main dataset. Including records with constant METs values of 1.00000, or users who only engage with this function, could bias the results if not properly segmented.

Instead, this table will be analyzed separately to identify more active user profiles and better understand exercise patterns tracked through this metric.

3.5 Final Data Merge

Finally, the tables df_fitbit, DailySleep, and DailyWeight_clean were merged using Id and ActivityDate.

```
Full_DailyActivity <- df_fitbit %>%
  left_join(DailySleep, by = c("Id", "ActivityDate" = "SleepDay")) %>%
  left_join(DailyWeight_clean, by =c("Id", "ActivityDate" = "Date"))
```

During this merge, several users were found to have activity data but no sleep records. Thus, NA values were preserved to indicate that the sleep-tracking function was not used on those days. These missing values will be helpful for understanding user behavior and engagement with the device.

4 Analyze

4.1 Features and User Profiles

We aim to identify which features are most commonly used by users and what type of activity profiles they have. To start, we will examine the usage of the steps, sleep, and weight tracking features within the time range covered by the dataset.

```
# Calculate the percentage of usage per user for each function
feature_usage <- Full_DailyActivity %>%
  group_by(Id) %>%
  summarise(
    total_days = n(),
    days_with_steps = sum(TotalSteps > 0, na.rm = TRUE),
    days_with_sleep = sum(!is.na(TotalMinutesAsleep)),
    days_with_weight = sum(!is.na(WeightKg))
  ) %>%
  mutate(
    pct_steps = days_with_steps / total_days * 100,
    pct_sleep = days_with_sleep / total_days * 100,
    pct_weight = days_with_weight / total_days * 100
  )

# Calculate overall average usage by function
average_usage <- feature_usage %>%
  summarise(
    Steps = mean(pct_steps),
    Sleep = mean(pct_sleep),
    Weight = mean(pct_weight)
  ) %>%
  pivot_longer(cols = everything(), names_to = "Feature", values_to = "Average_Usage")

head(average_usage)
```

```
## # A tibble: 3 x 2
##   Feature Average_Usage
##   <chr>           <dbl>
## 1 Steps            87.3
## 2 Sleep            40.8
## 3 Weight            6.85
```

During the observed period, the steps feature was the most used by users, showing significantly higher usage than the sleep and weight tracking functions.

This suggests that users:

Record steps almost daily, likely because the device tracks them automatically (87.3%).

Use the sleep tracking feature intermittently (40.8%).

Rarely log their body weight, which suggests that this feature requires extra effort due to manual input (6.8%).

4.2 Average Daily Steps (excluding zeros)

```
# We filter the days with steps greater than 0
steps_nonzero <- Full_DailyActivity %>%
  filter(TotalSteps > 0)

# We calculate general average
avg_steps <- mean(steps_nonzero$TotalSteps)

# We show the average
avg_steps
```

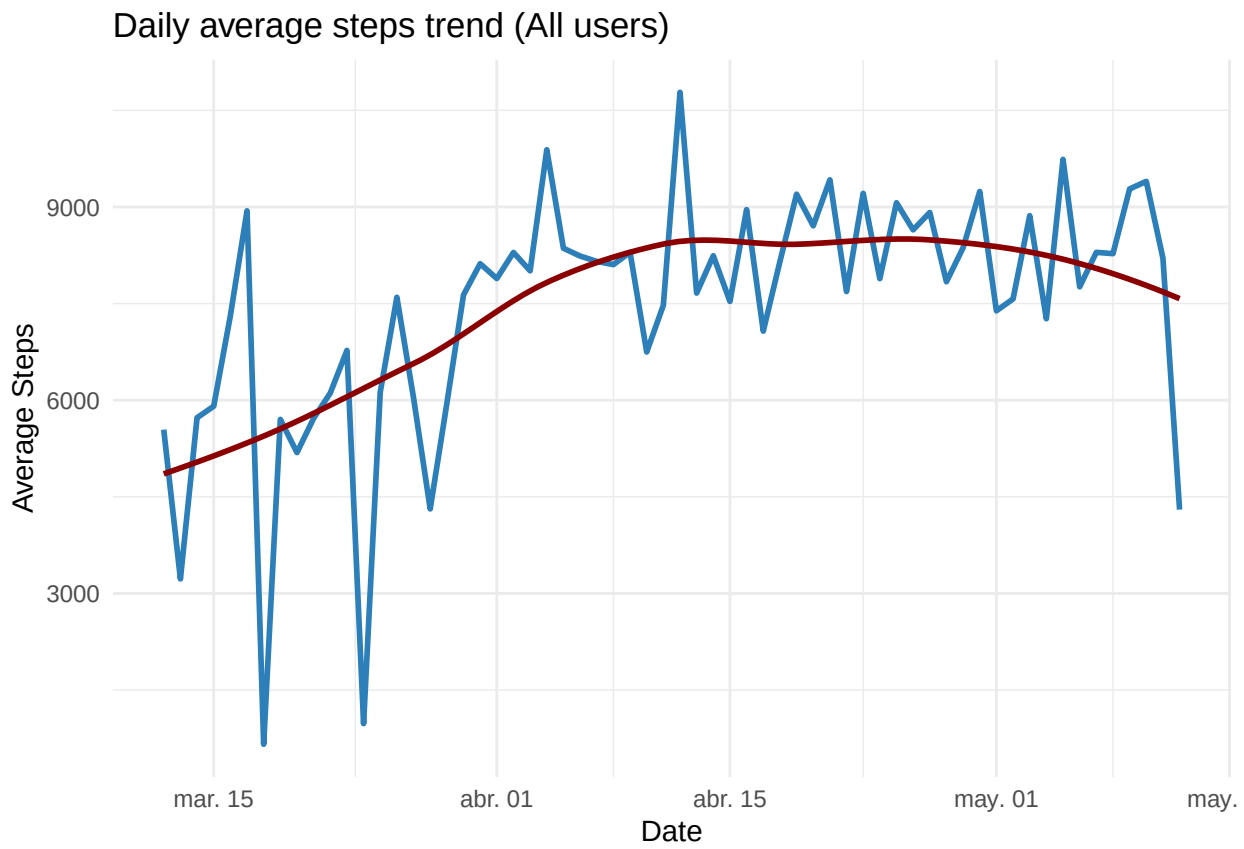
```
## [1] 8223.062
```

On days when users recorded steps (excluding zeros), the average was approximately 8,223 steps per day. While this figure is close to the commonly recommended 10,000 steps, it indicates that many users consistently fall short of this goal.

Next, we will analyze whether there is an upward trend in daily steps over time as users continue using the device.

```
Full_DailyActivity %>%
  filter(TotalSteps > 0) %>%
  group_by(ActivityDate) %>%
  summarise(avg_steps = mean(TotalSteps)) %>%
  ggplot(aes(x = ActivityDate, y = avg_steps)) +
  geom_line(color = "#2c7fb8", linewidth = 1) +
  geom_smooth(method = "loess", se = FALSE, color = "darkred") +
  labs(
    title = "Daily average steps trend (All users)",
    x = "Date",
    y = "Average Steps"
  ) +
  theme_minimal()
```

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



On average, users increase their step count during the first few weeks of using the device, reaching a peak in mid-April (approximately three weeks into usage). However, this trend does not persist over time: a decline in the average daily step count is observed afterward, suggesting a drop in motivation or adherence to device use.

4.3 User Sleep Analysis

```
# Calculate the average number of minutes slept per user (only where sleep data is available)
sleep_by_user <- Full_DailyActivity %>%
  filter(!is.na(TotalMinutesAsleep)) %>%
  group_by(Id) %>%
  summarise(avg_sleep_minutes = mean(TotalMinutesAsleep))

# Calculate the overall average sleep among all users
overall_avg_sleep <- mean(sleep_by_user$avg_sleep_minutes)

# Show the result
overall_avg_sleep
```

```
## [1] 371.0668
```

We observed that users sleep an average of 371 minutes, which is approximately 6 hours per night. This amount falls below the healthy recommended range. Therefore, we will analyze whether sleep duration increases as users continue using the device.

```r
# Convert date to number for linear regression
sleep_slope <- Full_DailyActivity %>%
  filter(!is.na(TotalMinutesAsleep)) %>%
  mutate(date_numeric = as.numeric(ActivityDate)) %>%
  group_by(Id) %>%
  filter(n() >= 10) %>%  # at least 10 days with sleep data
  do(tidy(lm(TotalMinutesAsleep ~ date_numeric, data = .))) %>%
  filter(term == "date_numeric") %>%
  rename(slope = estimate)

# See the average increase in users recording sleep data

sleep_slope %>%
  mutate(
    trend = ifelse(slope > 0, "Increasing", "Not increasing")
  ) %>%
  select(Id, slope, trend)
```

```
## # A tibble: 17 x 3
## # Groups:   Id [17]
##            Id   slope trend
##         <dbl>   <dbl> <chr>
##  1 1503960366  0.0671 Increasing
##  2 1927972279 -0.574  Not increasing
##  3 2026352035 -1.03   Not increasing
##  4 2347167796  0.842  Increasing
##  5 3977333714  0.299  Increasing
##  6 4020332650 -0.567  Not increasing
##  7 4319703577  0.104  Increasing
##  8 4388161847  2.00   Increasing
##  9 4445114986  0.593  Increasing
## 10 4702921684  0.746  Increasing
## 11 5553957443 -1.28   Not increasing
## 12 5577150313  0.347  Increasing
## 13 6117666160  3.23   Increasing
## 14 6962181067 -0.439  Not increasing
## 15 7086361926  0.714  Increasing
## 16 8378563200 -0.742  Not increasing
## 17 8792009665 -1.83   Not increasing
```

Although 59% of users showed a positive trend in their sleep habits, most of these increases were very small (less than 1 additional minute per day). Only 2 or 3 users demonstrated a clearly significant trend (slope $>$ 2), which means we cannot conclude that there is a generalized improvement in sleep duration as a result of device usage.

4.4 Users' Activity Profile

```r
activity_profile <- Full_DailyActivity %>%
  group_by(Id) %>%
  summarise(
    very_active = mean(VeryActiveMinutes),
```

```
    fairly_active = mean(FairlyActiveMinutes),
    lightly_active = mean(LightlyActiveMinutes),
    sedentary = mean(SedentaryMinutes)
  )

activity_profile <- activity_profile %>%
  mutate(
    total = very_active + fairly_active + lightly_active + sedentary,
    pct_very = very_active / total * 100,
    pct_fairly = fairly_active / total * 100,
    pct_lightly = lightly_active / total * 100,
    pct_sedentary = sedentary / total * 100
  )

activity_profile %>%
  summarise(
    Very = mean(very_active),
    Fairly = mean(fairly_active),
    Lightly = mean(lightly_active),
    Sedentary = mean(sedentary)
  )
```

```
## # A tibble: 1 x 4
##    Very Fairly Lightly Sedentary
##   <dbl>  <dbl>   <dbl>     <dbl>
## 1  19.2   15.0    185.     1021.
```

On average, users meet the minimum physical activity recommendations established by the WHO(https://www.who.int/news-room/fact-sheets/detail/physical-activity).

However, since most of the day remains sedentary (over 17 hours), and vigorous activity is right at the lower recommended threshold, not all users exhibit a fitness-oriented profile. Instead, the data reflects a pattern of moderate activity with high daily inactivity.

It is also worth noting that this data may not be fully representative, as there could be significant individual differences among users. Therefore, we will proceed to categorize users by their level of physical activity to determine the percentage that falls into each activity profile.

```
user_activity_type <- Full_DailyActivity %>%
  group_by(Id) %>%
  summarise(
    avg_very = mean(VeryActiveMinutes),
    avg_fairly = mean(FairlyActiveMinutes)
  ) %>%
  mutate(
    activity_group = case_when(
      avg_very >= 11 | (avg_very + avg_fairly) >= 21 ~ "Fitness",
      avg_very + avg_fairly > 5 ~ "Moderately Active",
      avg_very + avg_fairly > 0 ~ "Light Movers",
      TRUE ~ "Sedentary"
    )
  )
user_activity_type %>%
```

```
  count(activity_group) %>%
  arrange(desc(n))
```

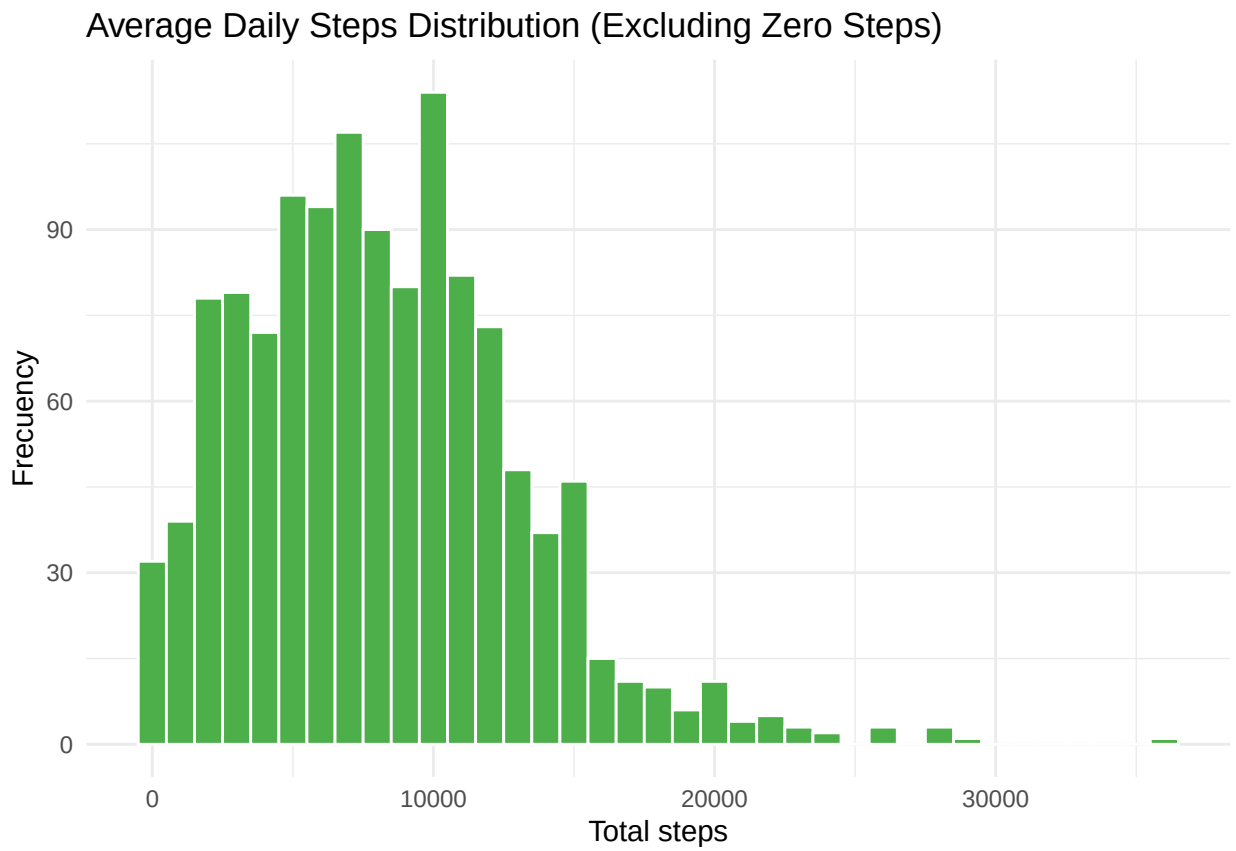
```
## # A tibble: 3 x 2
##   activity_group        n
##   <chr>             <int>
## 1 Fitness              20
## 2 Moderately Active    10
## 3 Light Movers          5
```

Here we observe that the majority of our users exercise regularly and could be considered as having a fitness-oriented profile.

5. Share

5.1 Distribution of Users' Daily Steps

```
ggplot(steps_nonzero, aes(x = TotalSteps)) +
  geom_histogram(binwidth = 1000, fill = "#4DAF4A", color = "white") +
  labs(
    title = "Average Daily Steps Distribution (Excluding Zero Steps)",
    x = "Total steps",
    y = "Frecuency"
  ) +
  theme_minimal()
```

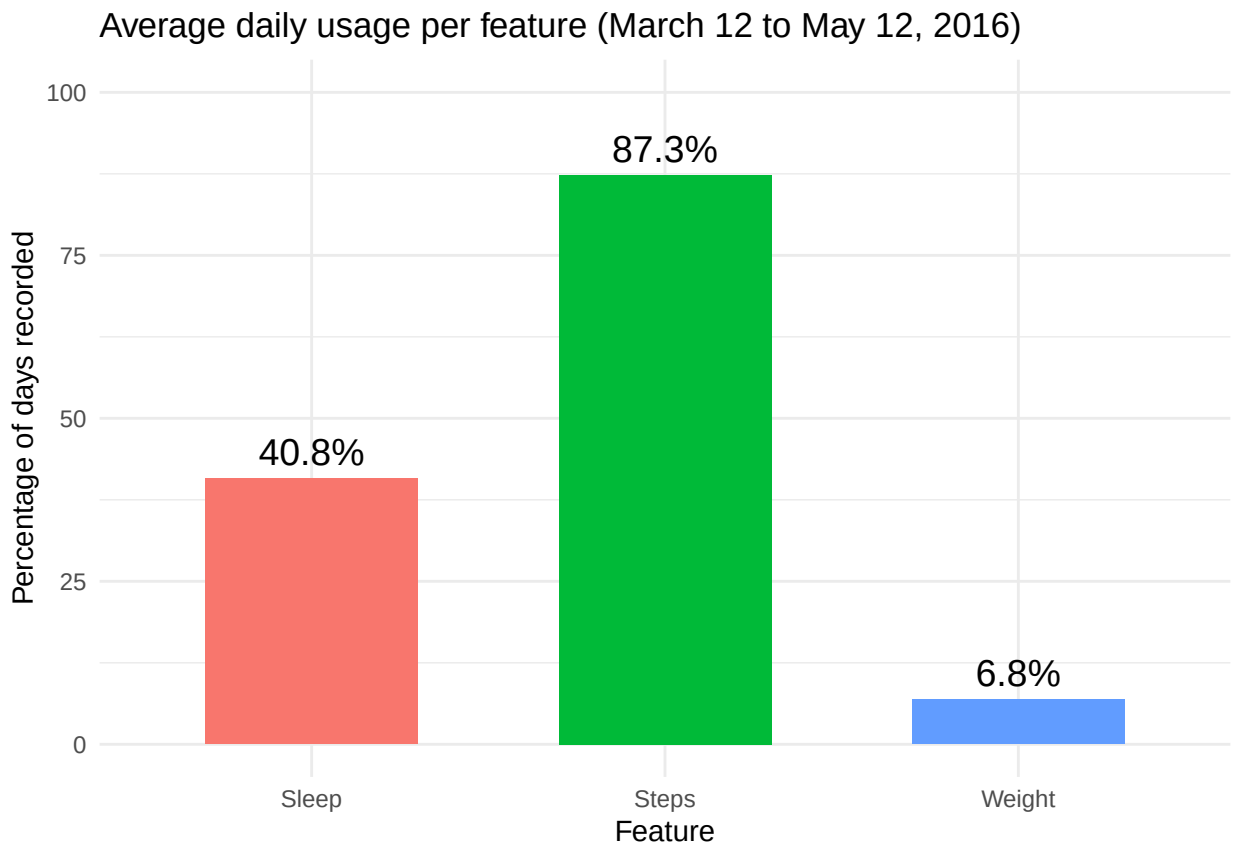


Most users take between 2,000 and 10,000 steps per day, with a small group reaching the recommended 10,000+ steps.

This behavior reflects a lifestyle ranging from slightly active to sedentary for the majority of users.

5.2 Device Feature Usage During the Observed Time Period

```
ggplot(average_usage, aes(x = Feature, y = Average_Usage, fill = Feature)) +
  geom_col(width = 0.6, show.legend = FALSE) +
  geom_text(aes(label = paste0(round(Average_Usage, 1), "%")),
            vjust = -0.5, size = 5) +
  scale_y_continuous(limits = c(0, 100)) +
  labs(
    title = "Average daily usage per feature (March 12 to May 12, 2016)",
    y = "Percentage of days recorded",
    x = "Feature"
  ) +
  theme_minimal()
```

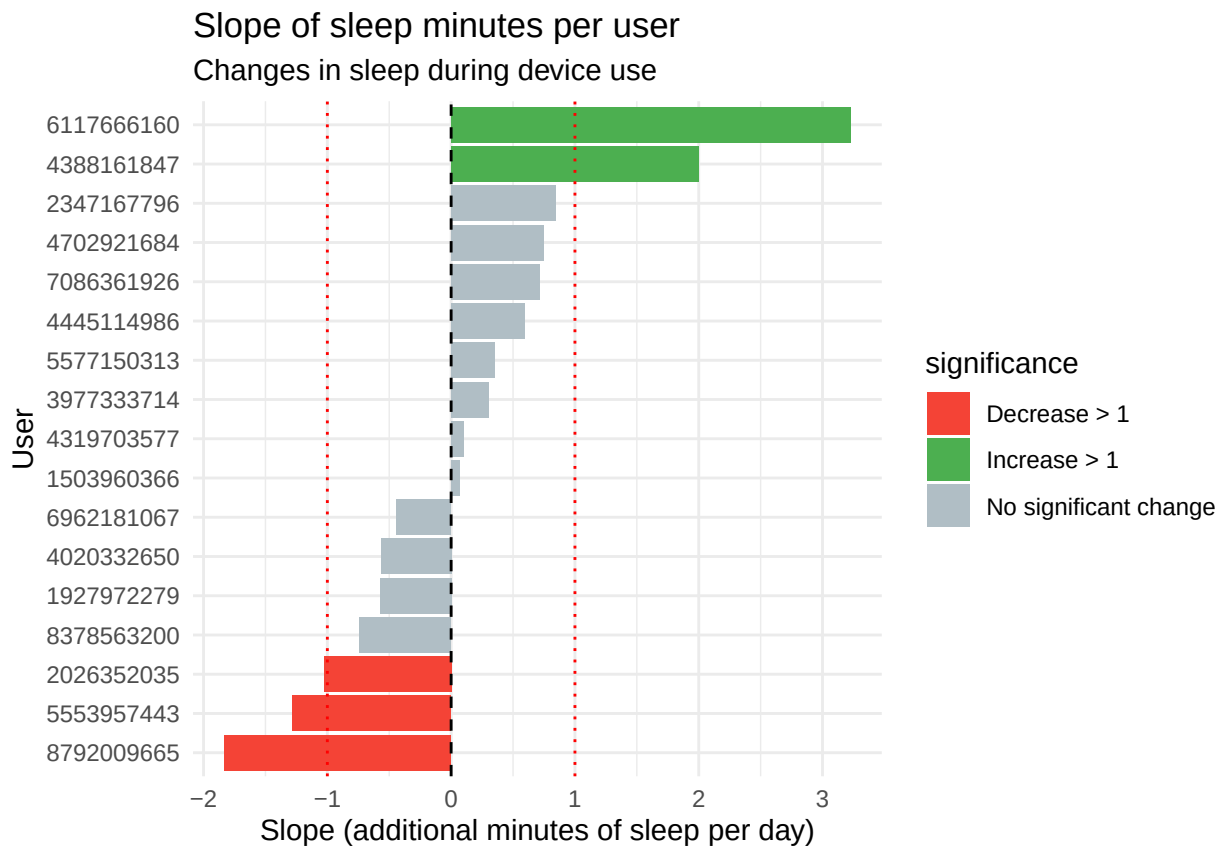


5.3 Slope of sleep minutes per user

```
# Sort by slope and create an interpretation variable
sleep_slope_plot <- sleep_slope %>%
  mutate(
    user = as.factor(Id),
    significance = case_when(
      slope > 1 ~ "Increase > 1",
      slope < -1 ~ "Decrease > 1",
      TRUE ~ "No significant change"
    )
  )

ggplot(sleep_slope_plot, aes(x = reorder(user, slope), y = slope, fill = significance)) +
  geom_col(show.legend = TRUE) +
  coord_flip() +
  geom_hline(yintercept = 0, color = "black", linetype = "dashed") +
  geom_hline(yintercept = 1, color = "red", linetype = "dotted") +
  geom_hline(yintercept = -1, color = "red", linetype = "dotted") +
  labs(
    title = "Slope of sleep minutes per user",
    subtitle = "Changes in sleep during device use",
    x = "User",
    y = "Slope (additional minutes of sleep per day)"
```

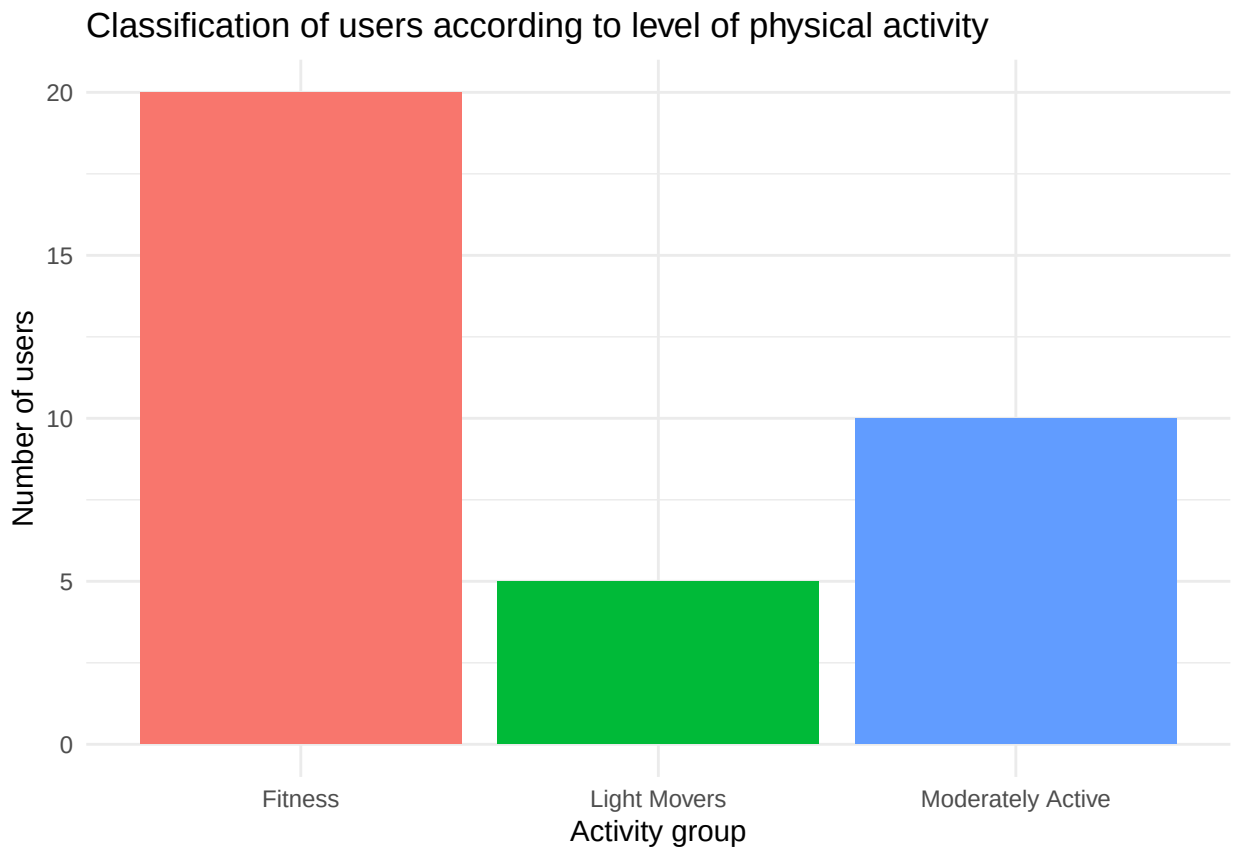
```
  ) +
  scale_fill_manual(values = c(
    "Increase > 1" = "#4CAF50",
    "Decrease > 1" = "#F44336",
    "No significant change" = "#B0BEC5"
  )) +
  theme_minimal()
```



This chart clearly shows that most users do not experience a significant change in sleep duration over time. Almost all remain within a neutral range, with only a few users showing notable improvements in their sleep patterns.

5.4 User Classification by Physical Activity Level

```
ggplot(user_activity_type, aes(x = activity_group, fill = activity_group)) +
  geom_bar(show.legend = FALSE) +
  labs(
    title = "Classification of users according to level of physical activity",
    x = "Activity group",
    y = "Number of users"
  ) +
  theme_minimal()
```



Once the users are grouped, we can determine that the majority of those who use the device fall into the Fitness or Moderately Active categories, with this group being four times larger than the group classified as more sedentary.

This suggests that users tend to use the device as a tool to support physical activity or sports practice. ## 6. Act

6.1 Key Findings

- Most users primarily use the step tracking feature, while the sleep and weight functions are used less frequently.
- Although there is an initial increase in physical activity during the first few weeks, this effect does not persist over time.
- The average sleep duration (6 hours) falls below the recommended threshold, and no significant improvements are observed with device usage.
- Users tend to fall into moderately active or fitness-oriented profiles, which are four times more common than sedentary profiles.
- This suggests that the device is used more as a support tool for physical activity rather than for comprehensive health monitoring.

6.2 Recommendations for the Marketing Team

- **Encourage long-term use of the device:** Since activity levels tend to drop after 3 weeks, consider sending motivational notifications or virtual rewards to maintain user engagement.

- **Promote the sleep tracking feature:** Create educational content about the importance of sleep and make the feature easier to activate automatically.
- **Launch campaigns segmented by user profile:**
  - *For fitness-oriented users:* Offer advanced training features and weekly challenges.
  - *For moderately active users:* Incentivize small daily achievements and self-progress comparisons.
- **Encourage weight tracking** by integrating with smart scales or providing rewards for consistent logging.