Case_Study_Bellabeat_Report

Daniel Marin

2025-07-21

1. Tarea Empresarial

El equipo de analistas de marketing de Bellabeat, una empresa de tecnología de vanguardia enfocada en la salud femenina, fue asignado por Urška Sršen para investigar cómo los usuarios utilizan los productos de la compañía. Sršen cree firmemente que los datos generados por estos dispositivos pueden revelar información valiosa que permita impulsar el crecimiento estratégico de la empresa.

Como analista junior del equipo, se me ha encomendado analizar uno de los productos de Bellabeat con el objetivo de comprender cómo interactúan los usuarios con él. La meta es aplicar estos hallazgos al perfil actual de clientes de la marca y proporcionar al equipo de marketing una base sólida de datos para respaldar futuras decisiones estratégicas.

Este análisis busca responder tres preguntas clave:

- ¿Qué tendencias existen en el uso de dispositivos inteligentes?
- ¿Cómo podrían estas tendencias aplicarse a las usuarias de Bellabeat?
- ¿Cómo podrían influir estas tendencias en la estrategia de marketing de la empresa?

2. Prepare: Data Description

Para este caso de estudio se usó un conjunto de datos públicos disponible en Kaggle, publicado por el usuario Arash Nic. Este dataset cuenta con una licencia CC0 (dominio público), lo que permite su uso libre para fines educativos y de análisis.

El archivo descargado fue un .zip que contenía múltiples archivos .csv organizados en carpetas según la fecha de exportación. Se identificaron dos carpetas principales: mturkfitbit_export_3.12.16 y mturkfitbit_export_4.12.16, ambas con archivos similares y continuidad temporal. Para el análisis, se decidió combinar ambas carpetas, validando que los datos del 4.12.16 comienzan donde terminan los del 3.12.16.

Los archivos están en formato wide, donde cada fila representa una observación por usuario y fecha. Los datos fueron almacenados localmente y cargados en R para su análisis.

Durante la exploración inicial se observó que no todos los usuarios están presentes en todos los archivos, probablemente porque cada usuario activó diferentes funciones del dispositivo Fitbit (como monitoreo de sueño, ritmo cardíaco, peso, entre otros). Este hallazgo se tuvo en cuenta para evitar uniones erróneas o pérdida de datos.

Con base en el criterio ROCCC, se determinó que los datos son:

Reliable (Confiables): recopilados directamente de dispositivos reales de 30 usuarios.

Original (Originales): sin modificaciones por terceros.

Comprehensive (Completos): incluyen datos por minuto, hora y día.

Current (Actuales): aunque son del año 2016, siguen siendo relevantes para explorar hábitos generales de uso de dispositivos inteligentes.

Cited (Citados): correctamente referenciados en Kaggle.

Por el momento no se utilizarán fuentes adicionales, aunque se contempla la posibilidad de incorporarlas más adelante para enriquecer el análisis.

3. Process: Limpieza y Transformación

3.1 Cargar paquetes y archivos

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

```
# Cargar archivos principales (de ambas carpetas)
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 Inspeccionar los datos

Inspeccionamos los datos que parecen más relevantes para responder las preguntas clave del proyecto. Notamos que en la carpeta del 3.12, la tabla de sueño (Sleep) proporciona información por minutos, mientras que en la carpeta del 4.12, los datos aparecen agregados por día. Para poder trabajar de forma uniforme, primero debemos estandarizar ambas tablas a formato diario. Después, fusionaremos los archivos de Daily Activity de ambas carpetas, y finalmente, unificaremos los datos de Sleep y Daily Activity en una sola tabla consolidada.

```
#Inspeccion de datos

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 Estandarización de los datos

Se inspeccionaron los archivos principales con head() para tener una visión general de su estructura y contenido.

Posteriormente, se estandarizaron las fechas de todas las tablas relevantes para poder trabajar con ellas de forma consistente. Algunas columnas contenían fechas en formato datetime, que fueron convertidas a fecha simple (Date) según la necesidad de cada análisis.

```r
#Estandarizacion de fechas para trabajar con ellas

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 Preparación de los datos

Datos de sueño Los registros por minuto de sueño fueron transformados a formato diario. Para ello, se agruparon por usuario y fecha, y se calcularon:

Total de minutos dormidos (cuando value == 1)

Total de minutos en cama (todas las observaciones)

Un registro diario por usuario (simulando TotalSleepRecords = 1)

Ambas tablas de sueño (por minutos y por días) fueron unificadas y los días solapados se consolidaron mediante suma, para evitar pérdida de datos.

```
# Agrupamos por usuario y fecha para calcular:
# - Total de minutos dormidos (value == 1)
# - Total de minutos en cama (todos los registros)
# - Número de registros (simulando TotalSleepRecords = 1 por día)

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

# Unimos ambas tablas
DailySleep <- bind_rows(df_SleepDay_3.12, df_SleepDay_4.12)

# Sumamos los datos de los dias que se solapan para evitar perdida de datos

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"
  )
```

Datos de METs Durante la exploración de la tabla minuteMETs, se detectaron valores anormalmente altos, de hasta 189 METs por minuto, lo cual es fisiológicamente imposible. Según el Compendium of Physical Activities, valores mayores a 20–23 METs por minuto no son realistas.

Deducción: los valores parecen estar escalados por un factor de 10. Al dividirlos entre 10, se alinean con rangos plausibles para actividades físicas intensas. Por lo tanto, se creó una nueva columna METs_scaled dividiendo los valores originales entre 10. Luego se calculó el promedio diario de METs por usuario.

```
# Asegúrate de tener METs ya escalados (divididos entre 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)

# Calcular promedio diario por usuario

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")

#Union de ambas tablas para consolidar DailyAvgMETs

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

#Limpieza de Datos solapados

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

Datos de peso Se unieron las tablas de registros de peso de ambas carpetas. Para simplificar el análisis posterior, se eliminaron columnas no necesarias como WeightPounds, Fat, BMI y LogId.

```
# Unimos ambas tablas de registros de Peso

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

# Eliminamos columnas que no hacen falta para unirla con la tabla que se creara mas adelante (Full_Dail

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

Datos de actividad diaria Las dos tablas dailyActivity se unificaron, se identificaron observaciones duplicadas por usuario y fecha, y se agruparon para consolidar los datos por día sin pérdida de información.

```r
#Previamente ya estandarizamos las fechas asi que ahora solo uniremos las tablas

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

#Verificamos los datos duplicados

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
#Y sumamos los datos duplicados para no perder informacion

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"
  )
```

En este punto se cuenta con las siguientes tablas principales:

df_fitbit (1373 observaciones)

DailyAvgMETs (1935 observaciones)

DailySleep (870 observaciones)

DailyWeight (100 observaciones)

Se detectó que la tabla DailyAvgMETs tiene más de 500 observaciones adicionales respecto a la tabla de actividad diaria. Tras comparar los usuarios y los rangos de fechas, se confirmó que coinciden. También se detectó que en varios días algunos usuarios presentan un valor promedio exacto de 1.00000 METs, lo cual sugiere que la función se ejecuta de forma automática incluso cuando no hay actividad relevante.

Por esta razón, se decidió no combinar la tabla DailyAvgMETs con el resto del conjunto principal (Full_DailyActivity) para evitar introducir ruido en los análisis generales. Esta tabla se analizará de forma independiente con fines exploratorios, especialmente para perfilar usuarios muy activos.

```
# Usuarios que están en METs pero no en 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
```

Verificamos que efectivamente se trata de los mismos usuarios y del mismo rango de fechas. También observamos que la tabla DailyAvgMETs presenta varios días en los que algunos usuarios registran un promedio exacto de 1.00000 METs, lo cual sugiere que los datos podrían estar siendo registrados de forma automática, incluso sin una actividad física real.

Estos días podrían interpretarse como días sin uso activo de la función METs. No obstante, para evitar introducir ruido en los análisis generales de comportamiento diario, se decidió no combinar esta tabla con el conjunto principal. Incluir registros con valores constantes de 1.00000 METs, o de usuarios que solo utilizan esta función, podría sesgar los resultados si no se segmentan adecuadamente.

En su lugar, esta tabla se analizará por separado con el fin de identificar perfiles de usuarios más activos y comprender mejor los patrones de ejercicio registrados mediante esta métrica.

3.5 Unificación final de datos

Finalmente, se unieron las tablas df_fitbit, DailySleep y DailyWeight_clean mediante uniones por Id y fecha (ActivityDate).

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

Durante esta combinación, se detectó que varios usuarios tienen registros de actividad pero no de sueño. Por lo tanto, se mantuvieron los valores NA como indicativos de que el monitoreo de sueño no fue utilizado esos días. Esta ausencia será útil para entender los patrones de uso del dispositivo y el compromiso del usuario.

4. Analisis

4.1 Funciones y perfil de usuarios

Queremos identificar qué funciones son las más utilizadas por los usuarios y qué tipo de perfil de actividad presentan. Para ello, comenzaremos revisando el uso de las funciones de pasos, sueño y peso dentro del rango de tiempo cubierto por los datos disponibles.

```
# Calcular el porcentaje de uso por usuario para cada función
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
  )

# Calcular promedio general de uso por función
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
```

Durante el período observado, la función de pasos fue la más utilizada por los usuarios, con un uso significativamente superior al de las funciones de sueño y peso.

Esto sugiere que los usuarios:

Registran pasos casi todos los días, probablemente porque el dispositivo lo hace de forma automática (87.3%).

Usan la función de sueño de manera intermitente (40.8%).

Raramente registran su peso corporal, lo cual indica que esta función requiere un esfuerzo adicional al implicar entrada manual (6.8%).

4.2 Promedio de pasos diarios (excluyendo 0)

```
# Filtramos los días con pasos mayores a 0
steps_nonzero <- Full_DailyActivity %>%
  filter(TotalSteps > 0)

# Calculamos promedio general
avg_steps <- mean(steps_nonzero$TotalSteps)

# Mostramos el promedio
avg_steps
```

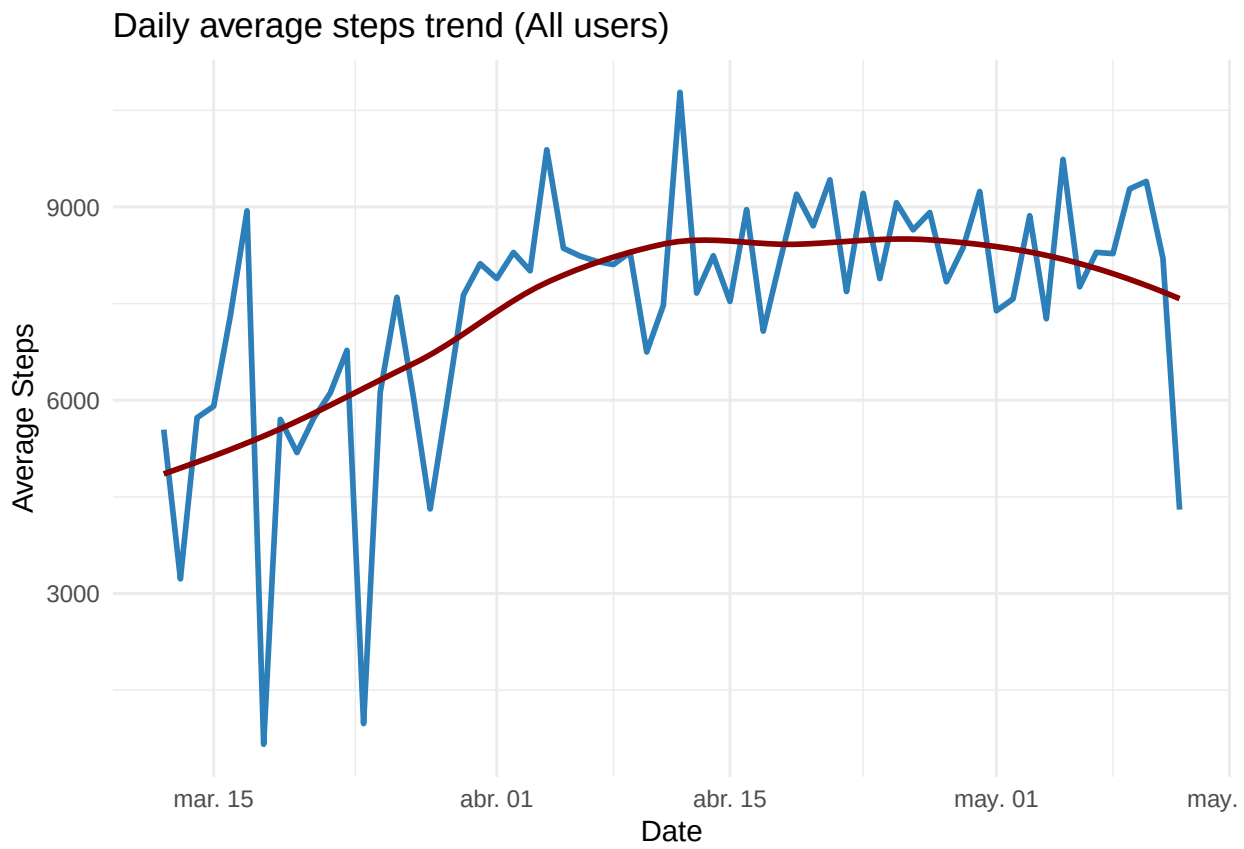
```
## [1] 8223.062
```

En los días en que los usuarios registraron pasos (excluyendo los valores de cero), el promedio diario fue de aproximadamente 8,223 pasos. Aunque este valor se aproxima al objetivo comúnmente recomendado de 10,000 pasos diarios, sugiere que muchos usuarios consistentemente no alcanzan esa meta.

A continuación, analizamos si existe una tendencia creciente en la cantidad de pasos conforme los usuarios utilizan el dispositivo durante el periodo observado.s

```
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'
```



En promedio, los usuarios aumentan su cantidad de pasos durante las primeras semanas de uso del dispositivo, alcanzando un punto máximo a mediados de abril (aproximadamente después de tres semanas de uso). Sin embargo, esta tendencia no se mantiene en el tiempo: posteriormente se observa una disminución en el promedio diario de pasos, lo cual sugiere una pérdida de motivación o adherencia al uso del dispositivo.

4.3 Análisis del sueño de los usuarios

```
# Calcular el promedio de minutos dormidos por usuario (solo donde hay datos de sueño)
sleep_by_user <- Full_DailyActivity %>%
  filter(!is.na(TotalMinutesAsleep)) %>%
  group_by(Id) %>%
  summarise(avg_sleep_minutes = mean(TotalMinutesAsleep))

# Calcular el promedio general de sueño entre todos los usuarios
overall_avg_sleep <- mean(sleep_by_user$avg_sleep_minutes)

# Mostrar el resultado
overall_avg_sleep
```

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

Observamos que los usuarios duermen, en promedio, 371 minutos, lo que equivale aproximadamente a 6 horas por noche. Esta cantidad se encuentra por debajo del rango considerado saludable. Por ello, analizaremos si la cantidad de sueño aumenta conforme los usuarios utilizan el dispositivo.

```r
# Convertir la fecha a número para regresión lineal
sleep_slope <- Full_DailyActivity %>%
  filter(!is.na(TotalMinutesAsleep)) %>%
  mutate(date_numeric = as.numeric(ActivityDate)) %>%
  group_by(Id) %>%
  filter(n() >= 10) %>%  # al menos 10 días con datos de sueño
  do(tidy(lm(TotalMinutesAsleep ~ date_numeric, data = .))) %>%
  filter(term == "date_numeric") %>%
  rename(slope = estimate)

# Ver el aumento promedio de los usuarios que registran datos de sueño

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
```

Aunque el 59% de los usuarios mostró una pendiente positiva en sus hábitos de sueño, la mayoría de estos incrementos fueron muy pequeños (menos de 1 minuto adicional por día). Solo 2 o 3 usuarios evidenciaron una tendencia claramente significativa (pendiente > 2), lo que no permite concluir que exista una mejora generalizada en la duración del sueño con el uso del dispositivo.

4.4 Perfil de actividad de los usuario

```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.
```

En promedio, los usuarios cumplen con las recomendaciones mínimas de actividad física establecidas por la OMS(https://www.who.int/news-room/fact-sheets/detail/physical-activity).

Sin embargo, dado que la mayor parte del día sigue siendo sedentaria (más de 17 horas), y que la actividad vigorosa se encuentra justo en el límite inferior recomendado, no todos los usuarios presentan un perfil fitness. Más bien, se observa un patrón de actividad moderada combinado con altos niveles de inactividad diaria.

También es importante señalar que estos datos podrían no ser completamente representativos, ya que pueden existir grandes diferencias individuales entre los usuarios. Por ello, se optará por diferenciar los tipos de usuarios según su nivel de actividad física, con el fin de determinar qué porcentaje pertenece a cada tipo de perfil.

```
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))
```

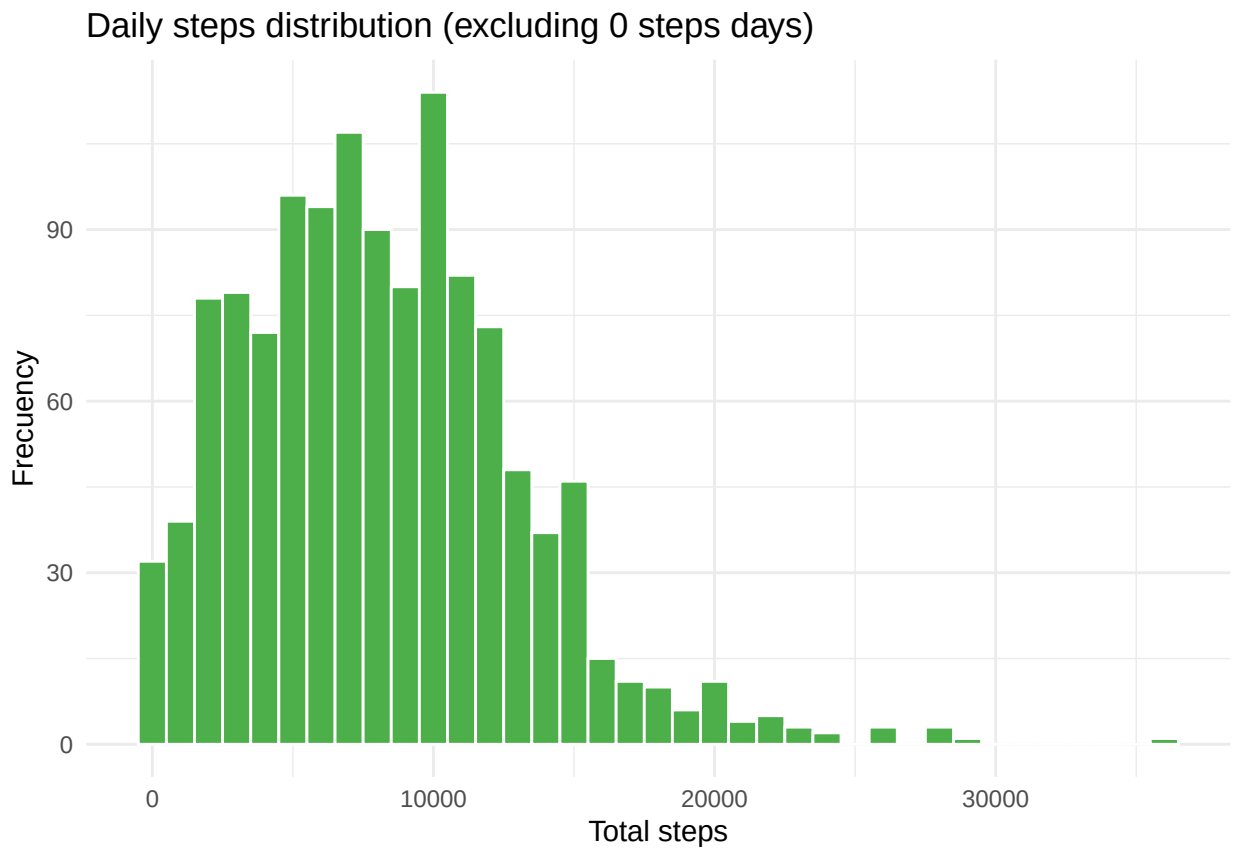
```
## # A tibble: 3 x 2
##   activity_group        n
##   <chr>             <int>
## 1 Fitness              20
## 2 Moderately Active    10
## 3 Light Movers          5
```

Aquí observamos que la mayoría de nuestros usuarios realiza ejercicio de forma regular y podrían considerarse como personas con un perfil Fitness.

5. Compartir

5.1 Distribucion de pasos diarios de los usuarios

```
ggplot(steps_nonzero, aes(x = TotalSteps)) +
  geom_histogram(binwidth = 1000, fill = "#4DAF4A", color = "white") +
  labs(
    title = "Daily steps distribution (excluding 0 steps days)",
    x = "Total steps",
    y = "Frecuency"
  ) +
  theme_minimal()
```

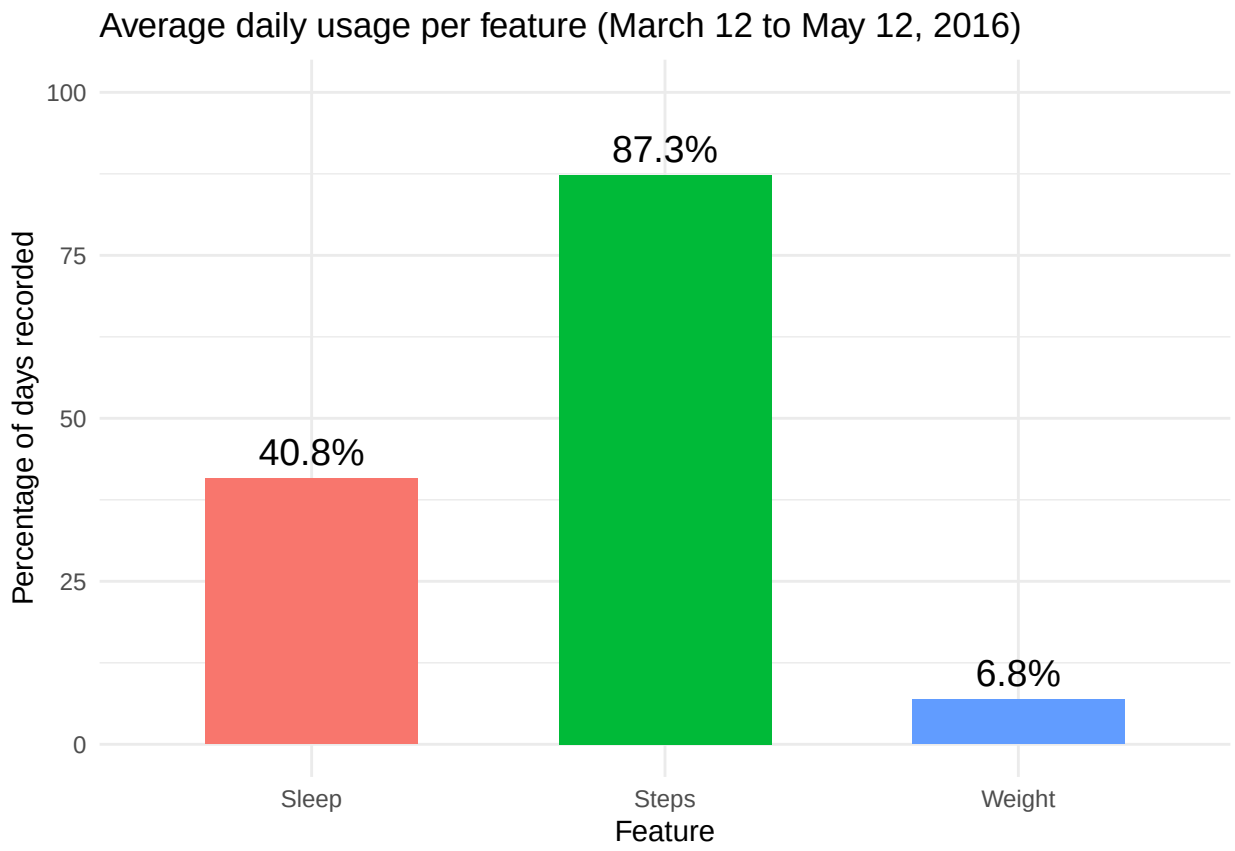


La mayoría de los usuarios da entre 2,000 y 10,000 pasos diarios, con un pequeño grupo que alcanza los más de 10,000 pasos recomendados.

Este comportamiento refleja un estilo de vida que va de ligeramente activo a sedentario en una gran parte de los usuarios.

5.2 Uso de funciones del dispositivo durante el período analizado

```
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()
```

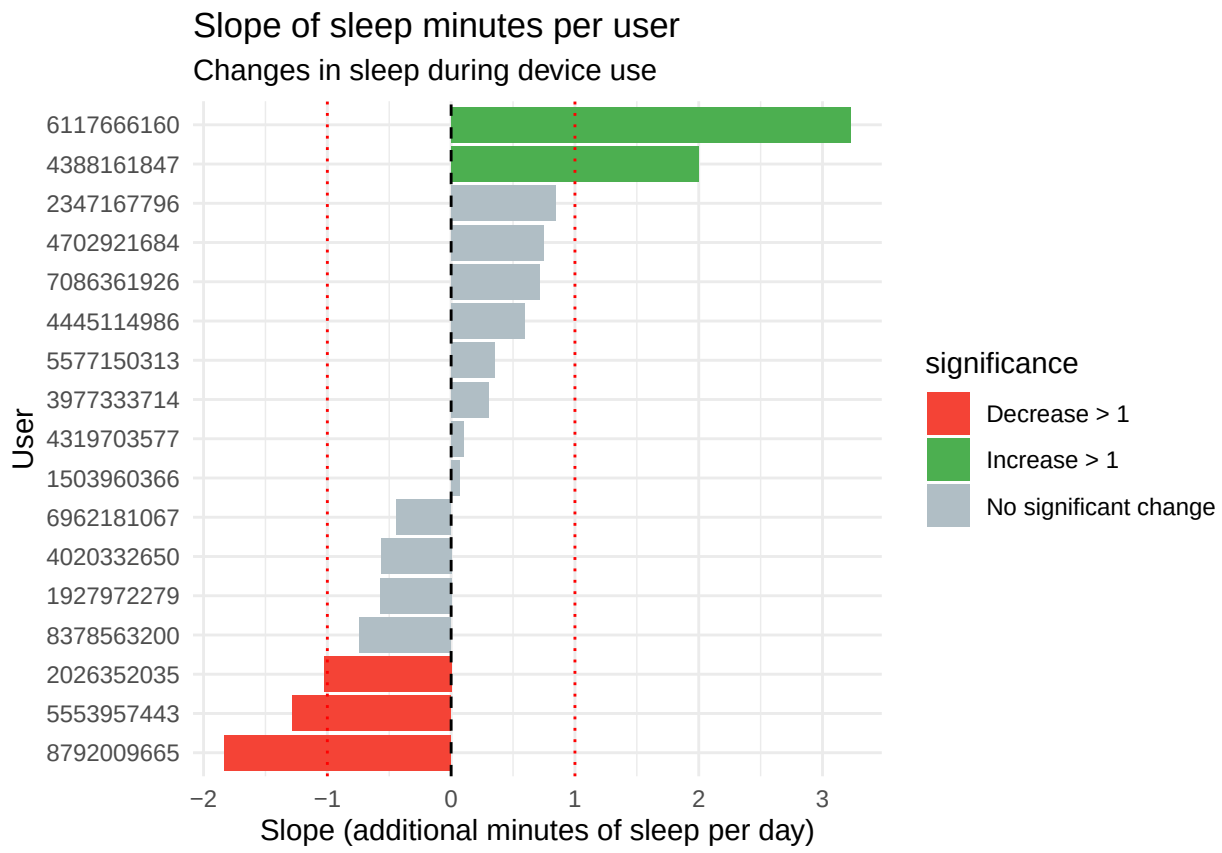


5.3 Pendiente de minutos minutos dormidos por usuario

```
# Ordenar por pendiente y crear una variable de interpretación
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)"
```

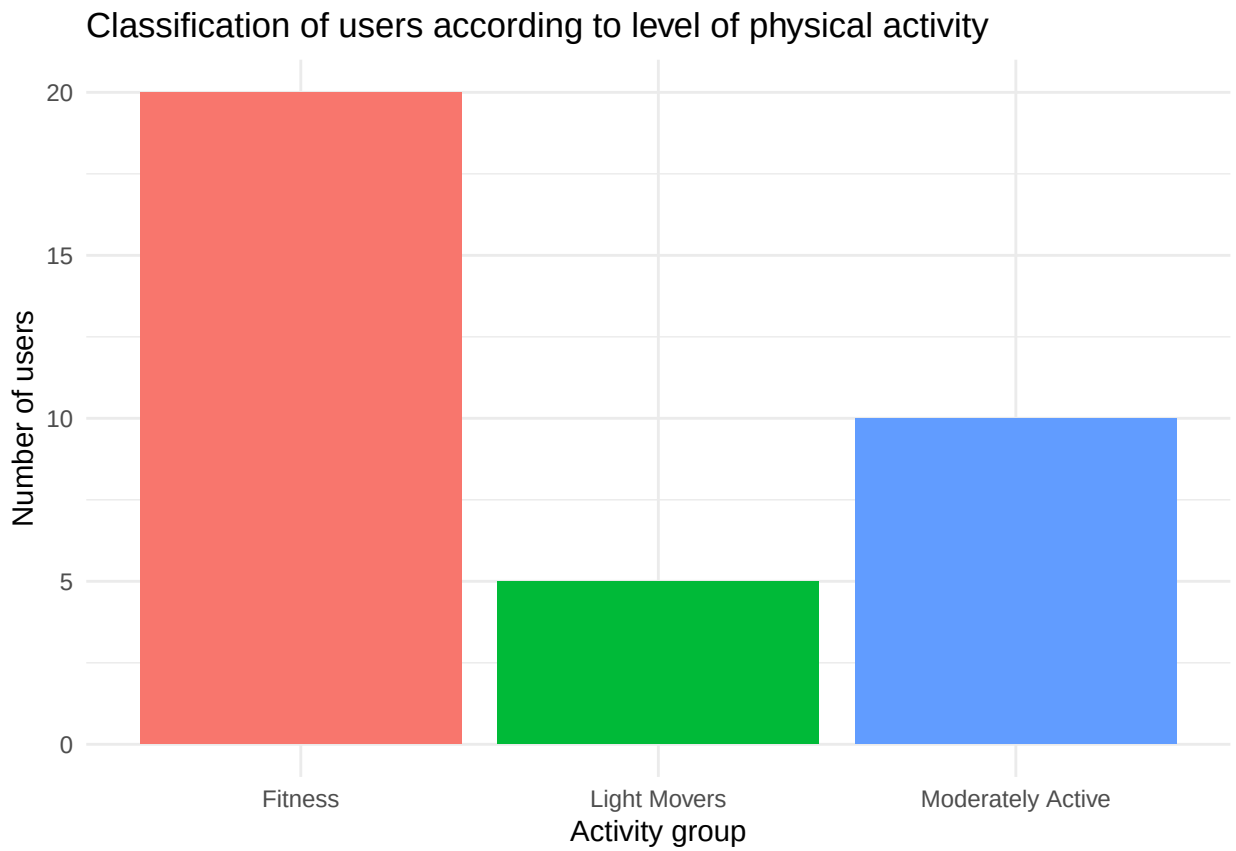
```
  ) +
  scale_fill_manual(values = c(
    "Increase > 1" = "#4CAF50",
    "Decrease > 1" = "#F44336",
    "No significant change" = "#B0BEC5"
  )) +
  theme_minimal()
```



Este gráfico deja en evidencia que la mayoría de los usuarios no experimenta un cambio significativo en la cantidad de sueño con el tiempo. Casi todos se mantienen dentro de un rango neutral, y solo un par de usuarios destacan con mejoras reales en sus hábitos de sueño.

5.4 Clasificación de usuarios según su nivel de actividad física

```
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()
```



Una vez agrupados los usuarios, podemos determinar que la mayoría de quienes utilizan el dispositivo son personas con un perfil Fitness o de actividad moderada, siendo este grupo cuatro veces más grande que el de los usuarios considerados más sedentarios.

Esto sugiere que los usuarios suelen emplear el dispositivo como una herramienta de apoyo para realizar actividad física o practicar deporte.

6. Conclusiones

6.1 Hallazgos descubiertos

- La mayoría de los usuarios utiliza principalmente la función de pasos, mientras que las funciones de sueño y peso son menos utilizadas.
- Aunque hay un aumento inicial en la actividad física durante las primeras semanas, este efecto no se mantiene en el tiempo.
- La duración del sueño promedio (6 horas) está por debajo de lo recomendado, y no se observan mejoras significativas con el uso del dispositivo.
- Los usuarios suelen tener perfiles moderadamente activos o fitness, siendo cuatro veces más numerosos que los perfiles sedentarios.
- Esto sugiere que el dispositivo se utiliza más como herramienta de apoyo para la actividad física que para el monitoreo integral de salud.

6.2 Recomendaciones para equipo de Marketing

- Fomentar el uso prolongado del dispositivo: Dado que la actividad tiende a disminuir después de 3 semanas, se podrían enviar notificaciones motivacionales o premios virtuales para mantener el interés.
- Promover la función de monitoreo de sueño: Crear contenido educativo sobre la importancia del sueño y facilitar su activación automática.
- Campañas segmentadas por perfil de usuario:
- Para usuarios fitness: Ofrecer funciones avanzadas de entrenamiento y retos semanales.
- Para perfiles moderados: Incentivar pequeños logros diarios y comparaciones con su propio progreso.
- Incentivar el registro de peso con integración automática a básculas inteligentes o recompensas por mantener registros consistentes.