Bellabeat Case Study

Prashant Yadav

2022-10-11

1. Introduction

Bellabeat, a high-tech manufacturer of health-focused products for women. Bellabeat is a successful small company, but they have the potential to become a larger player in the global smart device market. Urška Sršen, cofounder and Chief Creative Officer of Bellabeat, believes that analyzing smart device fitness data could help unlock new growth opportunities for the company.

1.2. About this Case Study

The primary goal of this case is to examine fitness data from smart devices and see how Bellabeat might benefit from new growth potential as a result. We will concentrate on the Bellabeat app, one of Bellabeat's products.The Bellabeat app provides users with health data related to their activity, sleep, stress, menstrual cycle, and mindfulness habits. This data can help users better understand their current habits and make healthy decisions. The Bellabeat app connects to their line of smart wellness products.

2. ASK Phase

2.1 Business Task

Determine some smart device usage trends using the Fitbit fitness tracker data. Explain how these trends might apply to Bellabeat's clients and how they might have an impact on the company's marketing plan. I'm hoping the questions below will lead me to the proper analysis.

A. What are some usage patterns, trends for smart devices?

B. How might Bellabeat customers be affected by these trends?

C. How can these developments affect Bellabeat's marketing plan?

2.2 Stakeholders

Urška Sršen: Bellabeat's cofounder and Chief Creative Officer

Sando Mur: Mathematician and Bellabeat's cofounder; key member of the Bellabeat executive team

Bellabeat marketing analytics team: A team of data analysts responsible for collecting, analyzing, and reporting data that helps guide Bellabeat's marketing strategy.

3. PREPARE Phase

3.1 Dataset Used

FitBit Fitness Tracker Data is the data source utilised for our case study. This dataset was made public through Mobius and is kept in Kaggle. 33 Fitbit users' personal fitness tracker data are included in this open source data set. Thirty-three eligible Fitbit users agreed to provide their personal tracker data, which included minute-level output for heart rate, sleep, and physical activity monitoring. It contains data on daily activity, steps, and heart rate that may be examined to learn more about users' routines.

3.2 Dataset details

The FitBit Fitness Tracker Dataset has 18 .csv files that were downloaded. 15 files in Long and 3 files in wide formats are provided in the dataset. 33 users are represented in the dataset that is used for the analysis below throughout a 31-day span.

The data is licenced under CC0: Public Domain, waiving the author's global copyright rights as well as any related or adjacent rights to the fullest extent permitted by law. Without seeking consent, the work may be duplicated, altered, disseminated, and performed—even for profit.

3.3 Limitations of Dataset

1. No Metadata such as information on location, lifestyle, weather, temperature, humidity etc available in the dataset as this would have provided a deeper context to the analyis.
2. Due to the sample size restriction and the lack of demographic data such as age, gender we may experience sampling bias. Given that Bellabeat focuses on making products for women, this important piece of information is lacking. The information gathered might not account for the variations in physiology and activity patterns among various demographic groups. We are aware that such information is subject to a stringent privacy policy, nonetheless.
3. The time frame for a more illuminating examination is actually somewhat short because the data only covered occurrences over a 31-day span. Seasonal changes have a significant impact on user behavior and lifestyle decisions. For instance, user exercise habits vary depending on the season.

Due to the above limitations with the dataset we will be giving our case an operational approach.

4. PROCESS Phase

4.1 Tools Used

All processing, analysis and visualization of data will be done in R Programming with R STUDIO

4.2 Installing packages and opening up libraries

Using the below packages that will help in the analysis

. tidyverse . here . skimr . janitor . lubridate . ggplot2 . ggpubr . ggrepel . scales . RcolorBrewer

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
```

```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(here)
```

```
## here() starts at /cloud/project
```

```
library(skimr)
library(janitor)
```

```
##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(ggplot2)
library(ggrepel)
library(scales)
```

```
##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
library(RColorBrewer)
library(waffle)
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

4.3 Importing Datasets

We will upload the datasets that will enable us to complete our business task given the datasets we currently have. We'll concentrate on the following datasets in our analysis:

a. daily activity

b. sleepday

c. hourlySteps

We won't take into account datasets Weight(8 Users) & Heartrate(7 Users) for this analysis due to the limited sample size.

```
daily_activity <- read_csv("/cloud/project/dailyActivity_merged.csv")
```

```
## 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.
```

```
daily_sleep <- read_csv("/cloud/project/sleepDay_merged.csv")
```

```
## 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.
```

```
hourly_steps <- read_csv("/cloud/project/hourlySteps_merged.csv")
```

```
## Rows: 22099 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ActivityHour
## dbl (2): Id, StepTotal
##
## 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.
```

4.4 Exploring a few key tables

To check summary of each column

```
head(daily_activity)
```

```
## # A tibble: 6 x 15
##       Id Activ~1 Total~2 Total~3 Track~4 Logge~5 VeryA~6 Moder~7 Light~8 Seden~9
##    <dbl> <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 1.50e9 4/12/2~   13162    8.5     8.5         0    1.88   0.550    6.06       0
## 2 1.50e9 4/13/2~   10735    6.97    6.97        0    1.57   0.690    4.71       0
## 3 1.50e9 4/14/2~   10460    6.74    6.74        0    2.44   0.400    3.91       0
## 4 1.50e9 4/15/2~    9762    6.28    6.28        0    2.14   1.26     2.83       0
## 5 1.50e9 4/16/2~   12669    8.16    8.16        0    2.71   0.410    5.04       0
## 6 1.50e9 4/17/2~    9705    6.48    6.48        0    3.19   0.780    2.51       0
## # ... with 5 more variables: VeryActiveMinutes <dbl>,
## #   FairlyActiveMinutes <dbl>, LightlyActiveMinutes <dbl>,
## #   SedentaryMinutes <dbl>, Calories <dbl>, and abbreviated variable names
## #   1: ActivityDate, 2: TotalSteps, 3: TotalDistance, 4: TrackerDistance,
## #   5: LoggedActivitiesDistance, 6: VeryActiveDistance,
## #   7: ModeratelyActiveDistance, 8: LightActiveDistance,
```

```
## #   9: SedentaryActiveDistance
```

```
str(daily_activity)
```

```
## spec_tbl_df [940 x 15] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id                       : num [1:940] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityDate             : chr [1:940] "4/12/2016" "4/13/2016" "4/14/2016" "4/15/2016" ...
##  $ TotalSteps               : num [1:940] 13162 10735 10460 9762 12669 ...
##  $ TotalDistance            : num [1:940] 8.5 6.97 6.74 6.28 8.16 ...
##  $ TrackerDistance          : num [1:940] 8.5 6.97 6.74 6.28 8.16 ...
##  $ LoggedActivitiesDistance: num [1:940] 0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveDistance       : num [1:940] 1.88 1.57 2.44 2.14 2.71 ...
##  $ ModeratelyActiveDistance: num [1:940] 0.55 0.69 0.4 1.26 0.41 ...
##  $ LightActiveDistance      : num [1:940] 6.06 4.71 3.91 2.83 5.04 ...
##  $ SedentaryActiveDistance : num [1:940] 0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveMinutes        : num [1:940] 25 21 30 29 36 38 42 50 28 19 ...
##  $ FairlyActiveMinutes      : num [1:940] 13 19 11 34 10 20 16 31 12 8 ...
##  $ LightlyActiveMinutes     : num [1:940] 328 217 181 209 221 164 233 264 205 211 ...
##  $ SedentaryMinutes         : num [1:940] 728 776 1218 726 773 ...
##  $ Calories                 : num [1:940] 1985 1797 1776 1745 1863 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityDate = col_character(),
##   ..   TotalSteps = col_double(),
##   ..   TotalDistance = col_double(),
##   ..   TrackerDistance = col_double(),
##   ..   LoggedActivitiesDistance = col_double(),
##   ..   VeryActiveDistance = col_double(),
##   ..   ModeratelyActiveDistance = col_double(),
##   ..   LightActiveDistance = col_double(),
##   ..   SedentaryActiveDistance = col_double(),
##   ..   VeryActiveMinutes = col_double(),
##   ..   FairlyActiveMinutes = col_double(),
##   ..   LightlyActiveMinutes = col_double(),
##   ..   SedentaryMinutes = col_double(),
##   ..   Calories = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
head(daily_sleep)
```

```
## # A tibble: 6 x 5
##           Id SleepDay              TotalSleepRecords TotalMinutesAsleep TotalT~1
##        <dbl> <chr>                             <dbl>              <dbl>    <dbl>
## 1 1503960366 4/12/2016 12:00:00 AM                 1                327      346
## 2 1503960366 4/13/2016 12:00:00 AM                 2                384      407
## 3 1503960366 4/15/2016 12:00:00 AM                 1                412      442
## 4 1503960366 4/16/2016 12:00:00 AM                 2                340      367
## 5 1503960366 4/17/2016 12:00:00 AM                 1                700      712
## 6 1503960366 4/19/2016 12:00:00 AM                 1                304      320
## # ... with abbreviated variable name 1: TotalTimeInBed
```

```
str(daily_sleep)
```

```
## spec_tbl_df [413 x 5] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
```

```
##  $ Id                : num [1:413] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ SleepDay          : chr [1:413] "4/12/2016 12:00:00 AM" "4/13/2016 12:00:00 AM" "4/15/2016 12:00:0
##  $ TotalSleepRecords : num [1:413] 1 2 1 2 1 1 1 1 1 1 ...
##  $ TotalMinutesAsleep: num [1:413] 327 384 412 340 700 304 360 325 361 430 ...
##  $ TotalTimeInBed    : num [1:413] 346 407 442 367 712 320 377 364 384 449 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   SleepDay = col_character(),
##   ..   TotalSleepRecords = col_double(),
##   ..   TotalMinutesAsleep = col_double(),
##   ..   TotalTimeInBed = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
head(hourly_steps)
```

```
## # A tibble: 6 x 3
##           Id ActivityHour          StepTotal
##        <dbl> <chr>                     <dbl>
## 1 1503960366 4/12/2016 12:00:00 AM       373
## 2 1503960366 4/12/2016 1:00:00 AM        160
## 3 1503960366 4/12/2016 2:00:00 AM        151
## 4 1503960366 4/12/2016 3:00:00 AM          0
## 5 1503960366 4/12/2016 4:00:00 AM          0
## 6 1503960366 4/12/2016 5:00:00 AM          0
```

```
str(hourly_steps)
```

```
## spec_tbl_df [22,099 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id          : num [1:22099] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityHour: chr [1:22099] "4/12/2016 12:00:00 AM" "4/12/2016 1:00:00 AM" "4/12/2016 2:00:00 AM"
##  $ StepTotal   : num [1:22099] 373 160 151 0 0 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityHour = col_character(),
##   ..   StepTotal = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

4.5 Cleaning and Formatting

Now that we are aware of our dataset structures, lets process them further to check for any errors or inconsistencies

4.5.1 Verifying the number of users

We want to confirm the number of unique users each data frame before we continue with our cleanup.

```
n_unique(daily_activity$Id)
```

```
## [1] 33
```

```
n_unique(daily_sleep$Id)
```

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

```
n_unique(hourly_steps$Id)
```

```
## [1] 33
```

4.5.2 Checking for duplicates

```
sum(duplicated(daily_activity))
```

```
## [1] 0
```

```
sum(duplicated(daily_sleep))
```

```
## [1] 3
```

```
sum(duplicated(hourly_steps))
```

```
## [1] 0
```

4.5.3 Removing duplicates and N/A values

Few duplicates were observed which needs to be removed along with cleaning of any N/A values

```
daily_activity <- daily_activity %>%
  distinct() %>%
  drop_na()

daily_sleep <- daily_sleep %>%
  distinct() %>%
  drop_na()

hourly_steps <- hourly_steps %>%
  distinct() %>%
  drop_na()
```

Verify again for any duplicate entries

```
sum(duplicated(daily_activity))
```

```
## [1] 0
```

```
sum(duplicated(daily_sleep))
```

```
## [1] 0
```

```
sum(duplicated(hourly_steps))
```

```
## [1] 0
```

4.5.4 Cleaning date-time format for consistencies

Since we will be merging both data frames, we will concentrate on cleaning the date-time format for daily activity and daily sleep. We are using as_date rather than as_datetime because we can ignore the time on the daily sleep data frame.

```
daily_activity <- daily_activity %>%
  rename(date = ActivityDate) %>%
  mutate(date = as_date(date, format = "%m/%d/%Y"))

daily_sleep <- daily_sleep %>%
```

```
  rename(date = SleepDay) %>%
  mutate(date = as_date(date, format = "%m/%d/%Y %I:%M:%S %p", tz=Sys.timezone()))
```

```
## Warning: `tz` argument is ignored by `as_date()`
```

Checking cleaned datasets

```
head(daily_activity)
```

```
## # A tibble: 6 x 15
##           Id date       TotalS~1 Total~2 Track~3 Logge~4 VeryA~5 Moder~6 Light~7
##        <dbl> <date>        <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 1503960366 2016-04-12    13162    8.5     8.5       0    1.88   0.550    6.06
## 2 1503960366 2016-04-13    10735    6.97    6.97      0    1.57   0.690    4.71
## 3 1503960366 2016-04-14    10460    6.74    6.74      0    2.44   0.400    3.91
## 4 1503960366 2016-04-15     9762    6.28    6.28      0    2.14   1.26     2.83
## 5 1503960366 2016-04-16    12669    8.16    8.16      0    2.71   0.410    5.04
## 6 1503960366 2016-04-17     9705    6.48    6.48      0    3.19   0.780    2.51
## # ... with 6 more variables: SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>, and
## #   abbreviated variable names 1: TotalSteps, 2: TotalDistance,
## #   3: TrackerDistance, 4: LoggedActivitiesDistance, 5: VeryActiveDistance,
## #   6: ModeratelyActiveDistance, 7: LightActiveDistance
```

```
head(daily_sleep)
```

```
## # A tibble: 6 x 5
##           Id date       TotalSleepRecords TotalMinutesAsleep TotalTimeInBed
##        <dbl> <date>                 <dbl>              <dbl>          <dbl>
## 1 1503960366 2016-04-12                 1                327            346
## 2 1503960366 2016-04-13                 2                384            407
## 3 1503960366 2016-04-15                 1                412            442
## 4 1503960366 2016-04-16                 2                340            367
## 5 1503960366 2016-04-17                 1                700            712
## 6 1503960366 2016-04-19                 1                304            320
```

Next, converting date string to date_time for hourly_steps dataset

```
hourly_steps <- hourly_steps %>%
  rename(date_time = ActivityHour) %>%
  mutate(date_time = as.POSIXct(date_time,format = "%m/%d/%Y %I:%M:%S %p" , tz=Sys.timezone()))

head(hourly_steps)
```

```
## # A tibble: 6 x 3
##           Id date_time           StepTotal
##        <dbl> <dttm>                  <dbl>
## 1 1503960366 2016-04-12 00:00:00       373
## 2 1503960366 2016-04-12 01:00:00       160
## 3 1503960366 2016-04-12 02:00:00       151
## 4 1503960366 2016-04-12 03:00:00         0
## 5 1503960366 2016-04-12 04:00:00         0
## 6 1503960366 2016-04-12 05:00:00         0
```

4.6 Merging Datasets

Using Id and date as primary keys for merging daily_activity and daily_sleep to check the correlation between variables

```
daily_activity_sleep <- merge(daily_activity, daily_sleep, by=c ("Id","date"))

glimpse(daily_activity_sleep)
```

```
## Rows: 410
## Columns: 18
## $ Id                       <dbl> 1503960366, 1503960366, 1503960366, 150396036~
## $ date                     <date> 2016-04-12, 2016-04-13, 2016-04-15, 2016-04-~
## $ TotalSteps               <dbl> 13162, 10735, 9762, 12669, 9705, 15506, 10544~
## $ TotalDistance            <dbl> 8.50, 6.97, 6.28, 8.16, 6.48, 9.88, 6.68, 6.3~
## $ TrackerDistance          <dbl> 8.50, 6.97, 6.28, 8.16, 6.48, 9.88, 6.68, 6.3~
## $ LoggedActivitiesDistance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ VeryActiveDistance       <dbl> 1.88, 1.57, 2.14, 2.71, 3.19, 3.53, 1.96, 1.3~
## $ ModeratelyActiveDistance <dbl> 0.55, 0.69, 1.26, 0.41, 0.78, 1.32, 0.48, 0.3~
## $ LightActiveDistance      <dbl> 6.06, 4.71, 2.83, 5.04, 2.51, 5.03, 4.24, 4.6~
## $ SedentaryActiveDistance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ VeryActiveMinutes        <dbl> 25, 21, 29, 36, 38, 50, 28, 19, 41, 39, 73, 3~
## $ FairlyActiveMinutes      <dbl> 13, 19, 34, 10, 20, 31, 12, 8, 21, 5, 14, 23,~
## $ LightlyActiveMinutes     <dbl> 328, 217, 209, 221, 164, 264, 205, 211, 262, ~
## $ SedentaryMinutes         <dbl> 728, 776, 726, 773, 539, 775, 818, 838, 732, ~
## $ Calories                 <dbl> 1985, 1797, 1745, 1863, 1728, 2035, 1786, 177~
## $ TotalSleepRecords        <dbl> 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ TotalMinutesAsleep       <dbl> 327, 384, 412, 340, 700, 304, 360, 325, 361, ~
## $ TotalTimeInBed           <dbl> 346, 407, 442, 367, 712, 320, 377, 364, 384, ~
```

5. ANALYZE & SHARE Phase

5.1 User type as per their activity levels

Since our sample doesn't contain any demographic information, we want to use the information we do have to identify the different types of consumers. Based on their daily step count, we may categories the users according to their activities. The following categories apply to users.

A. Sedentary - Less than 5000 steps/day

B. Lightly Active - Between 5000 - 7499 steps/day

C. Active - Between 7500 - 9999 steps/day

D. Highy Active - More than 10000 steps/day

First lets calculate the daily steps average by user

```
daily_average <- daily_activity_sleep %>%
  group_by(Id) %>%
  summarise(mean_daily_steps = mean(TotalSteps), mean_daily_calories = mean(Calories), mean_daily_sleep

head(daily_average)
```

```
## # A tibble: 6 x 4
##           Id mean_daily_steps mean_daily_calories mean_daily_sleep
##        <dbl>            <dbl>               <dbl>            <dbl>
## 1 1503960366           12406.               1872.             360.
## 2 1644430081            7968.               2978.             294
```

```
## 3 1844505072            3477                1676.             652
## 4 1927972279            1490                2316.             417
## 5 2026352035            5619.               1541.             506.
## 6 2320127002            5079                1804               61
```

Classifying users by daily average steps.

```
user_type <- daily_average %>%
  mutate(user_type = case_when(
    mean_daily_steps < 5000 ~ "Sedentary",
    mean_daily_steps >= 5000 & mean_daily_steps < 7499 ~ "Lightly Active",
    mean_daily_steps >= 7500 & mean_daily_steps < 9999 ~ "Active",
    mean_daily_steps >= 10000 ~ "Highly Active"
  ))

head(user_type)
```

```
## # A tibble: 6 x 5
##           Id mean_daily_steps mean_daily_calories mean_daily_sleep user_type
##        <dbl>            <dbl>               <dbl>            <dbl> <chr>
## 1 1503960366           12406.               1872.             360. Highly Active
## 2 1644430081            7968.               2978.             294  Active
## 3 1844505072            3477                1676.             652  Sedentary
## 4 1927972279            1490                2316.             417  Sedentary
## 5 2026352035            5619.               1541.             506. Lightly Acti~
## 6 2320127002            5079                1804               61  Lightly Acti~
```

Creating a dataframe with percentage for each user type for better visualization.

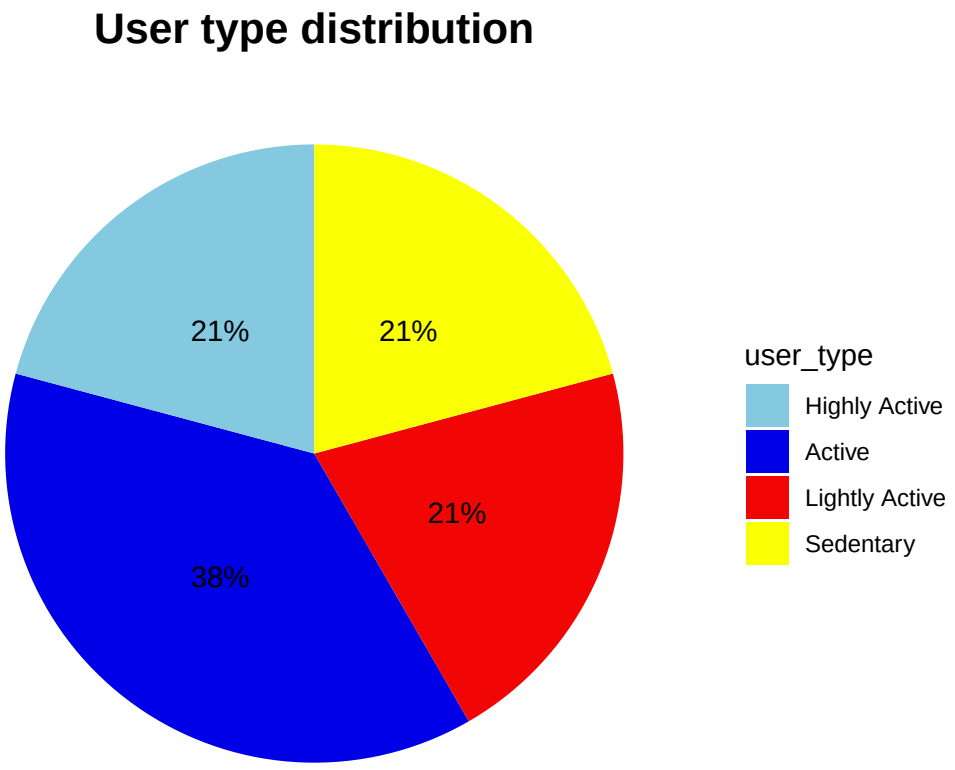
```
user_type_percent <- user_type %>%
  group_by(user_type) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(user_type) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))

user_type_percent$user_type <- factor(user_type_percent$user_type , levels = c("Highly Active","Active"


head(user_type_percent)
```

```
## # A tibble: 4 x 3
##   user_type      total_percent labels
##   <fct>                  <dbl> <chr>
## 1 Active                 0.375 38%
## 2 Highly Active          0.208 21%
## 3 Lightly Active         0.208 21%
## 4 Sedentary              0.208 21%
```

Below, we can see that users are fairly distributed according to their activity when daily step count is taken into account. Based on user activity, we can conclude that all users wear smart gadgets.

```
user_type_percent %>%
  ggplot(aes(x="",y=total_percent, fill=user_type)) +
  geom_bar(stat = "identity", width = 1)+
  coord_polar("y", start=0)+
```

```
  theme_minimal()+
  theme(axis.title.x= element_blank(),
        axis.title.y = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        plot.title = element_text(hjust = 0.5, size=16, face = "bold")) +
  scale_fill_manual(values = c("#85c9e0","#0000e6", "#f20505", "#fbff03")) +
  geom_text(aes(label = labels),
            position = position_stack(vjust = 0.5))+
  labs(title="User type distribution")
```



5.2 Steps walked & Asleep time on weekdays

We're interested in learning which days of the week users are busier and which days of the week they sleep more.

We'll also make sure the users are getting the required number of steps and hours of sleep.

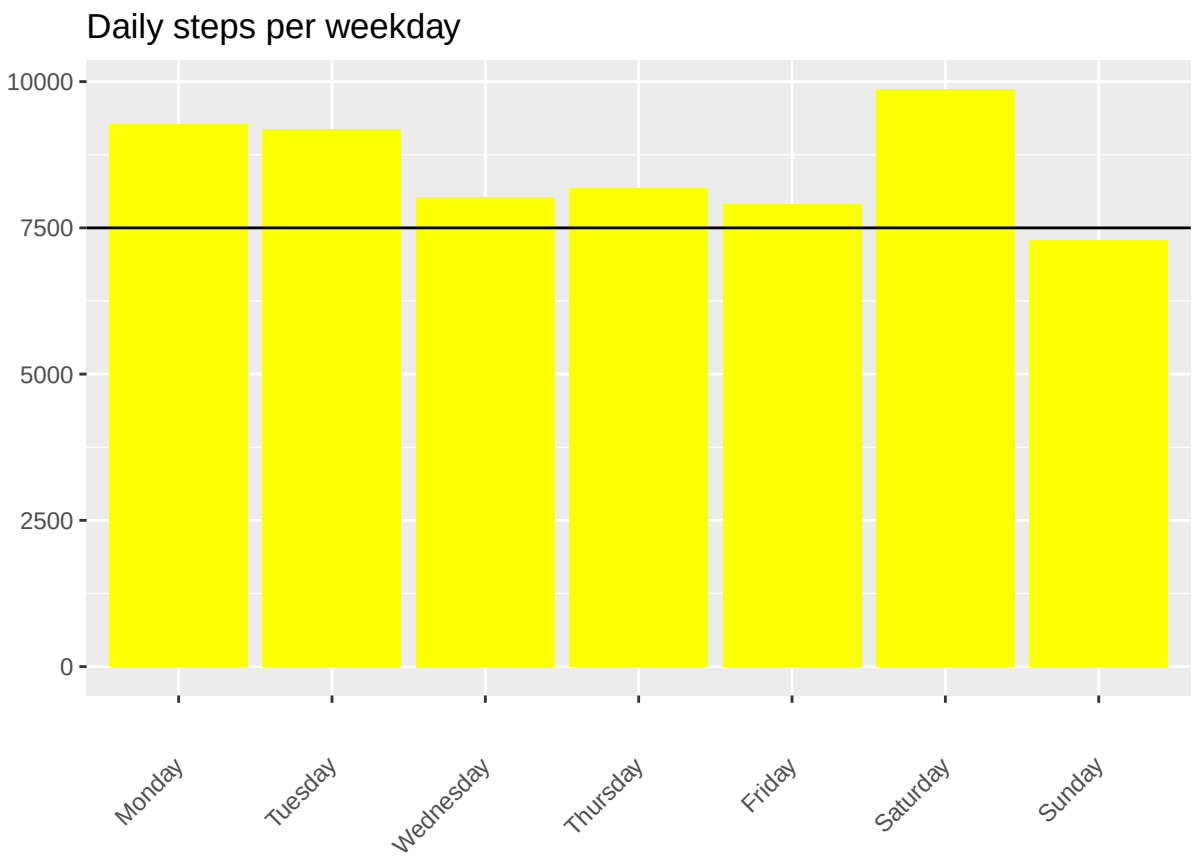
Based on the date in our column, we are calculating the weekdays below. We are also calculating the typical weekday steps and sleep duration.

```
weekday_steps_sleep <- daily_activity_sleep %>%
  mutate(weekday = weekdays(date))

weekday_steps_sleep$weekday <-ordered(weekday_steps_sleep$weekday, levels=c("Monday", "Tuesday", "Wednes
"Friday", "Saturday", "Sunday"))

 weekday_steps_sleep <-weekday_steps_sleep%>%
  group_by(weekday) %>%
```

```
  summarize (daily_steps = mean(TotalSteps), daily_sleep = mean(TotalMinutesAsleep))

head(weekday_steps_sleep)
```

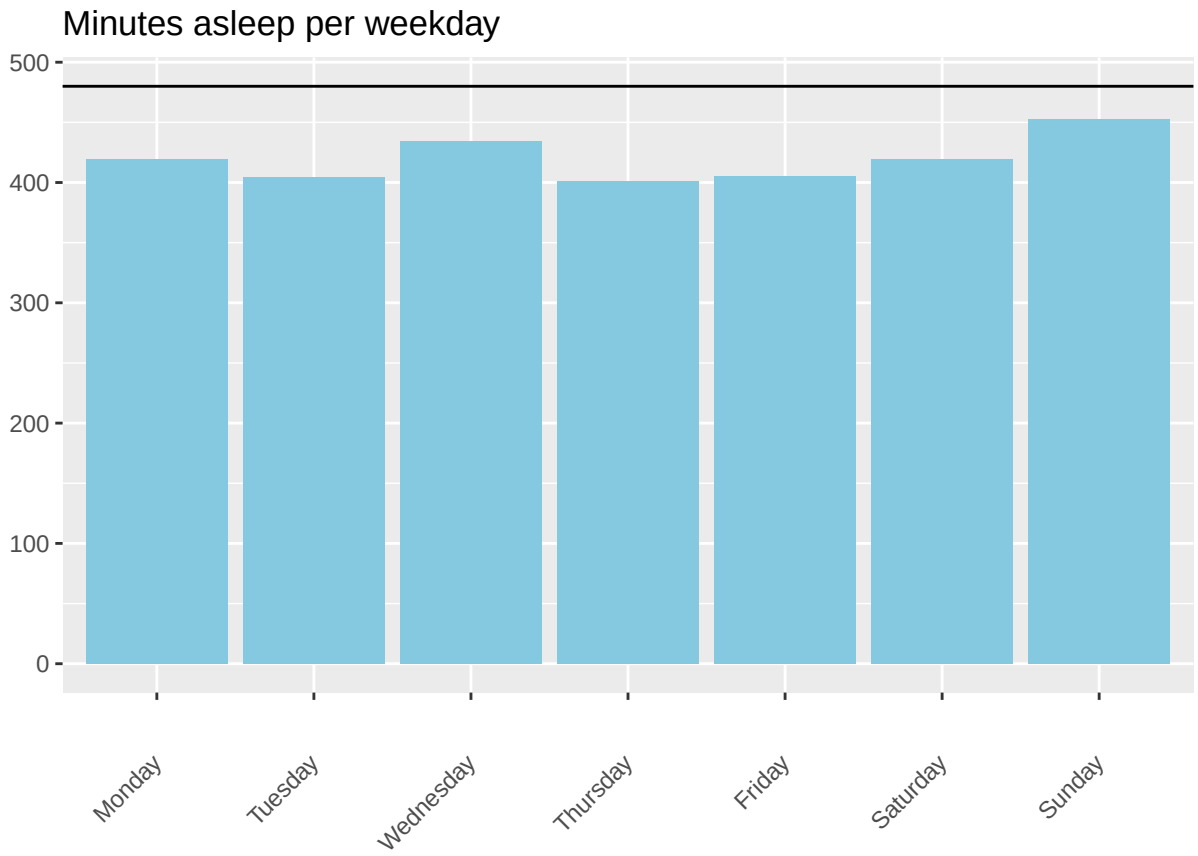
```
## # A tibble: 6 x 3
##   weekday   daily_steps daily_sleep
##   <ord>           <dbl>       <dbl>
## 1 Monday          9273.        420.
## 2 Tuesday         9183.        405.
## 3 Wednesday       8023.        435.
## 4 Thursday        8184.        401.
## 5 Friday          7901.        405.
## 6 Saturday        9871.        419.
```

```
ggplot(weekday_steps_sleep) +
      geom_col(aes(weekday, daily_steps), fill = "#fbff03") +
      geom_hline(yintercept = 7500) +
      labs(title = "Daily steps per weekday", x= "", y = "") +
      theme(axis.text.x = element_text(angle = 45,vjust = 0.5, hjust = 1))
```



. Users walk the daily recommended amount of steps of 7500 except on Sundays.

```
ggplot(weekday_steps_sleep, aes(weekday, daily_sleep)) +
      geom_col(fill = "#85c9e0") +
      geom_hline(yintercept = 480) +
      labs(title = "Minutes asleep per weekday", x= "", y = "") +
      theme(axis.text.x = element_text(angle = 45,vjust = 0.5, hjust = 1))
```



. Users don't sleep the 8 hours of sleep / night that is advised.

5.3 Hourly Steps/Day

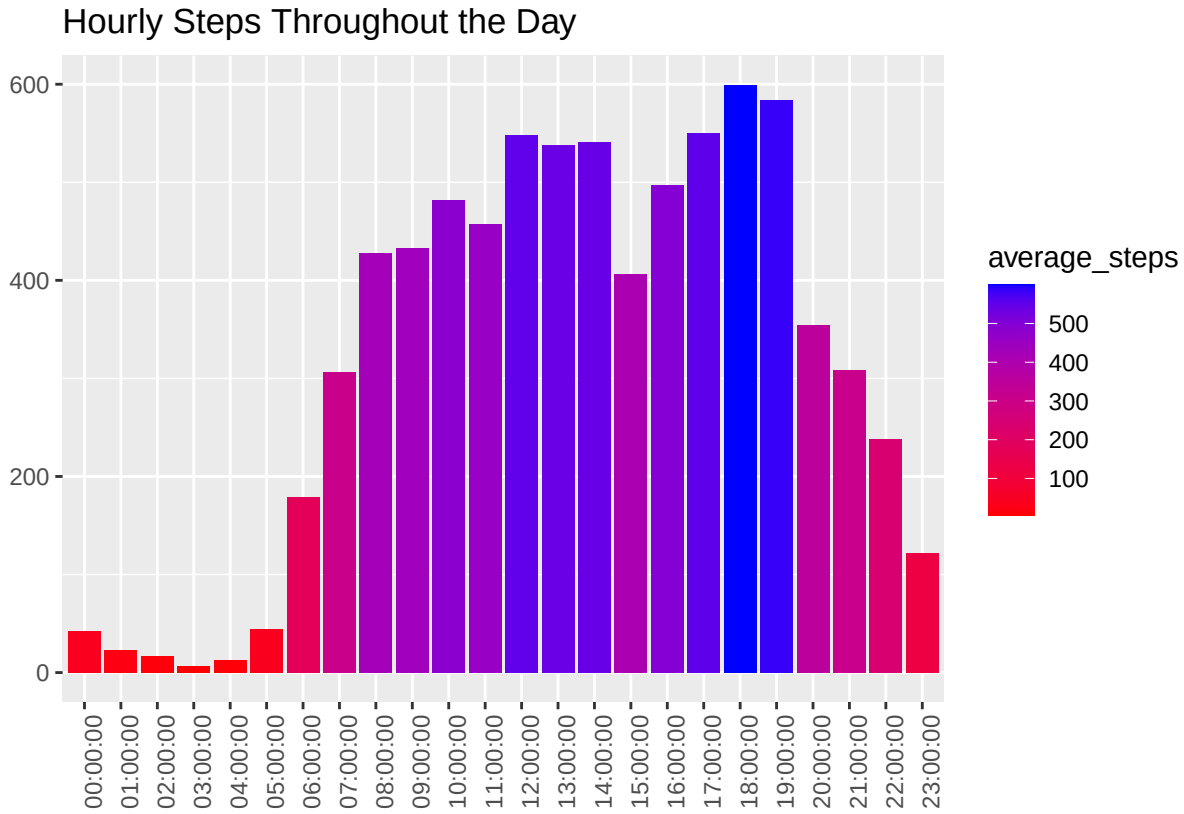
As we dig deeper into our analysis lets analyze user activity throughout the day and for the same we will use the hourly_steps data frame alongwith a separate date_time column.

```
hourly_steps <- hourly_steps %>%
  separate(date_time, into = c("date", "time"), sep= " ") %>%
  mutate(date = ymd(date))

head(hourly_steps)
```

```
## # A tibble: 6 x 4
##           Id date       time     StepTotal
##        <dbl> <date>     <chr>        <dbl>
## 1 1503960366 2016-04-12 00:00:00       373
## 2 1503960366 2016-04-12 01:00:00       160
## 3 1503960366 2016-04-12 02:00:00       151
## 4 1503960366 2016-04-12 03:00:00         0
## 5 1503960366 2016-04-12 04:00:00         0
## 6 1503960366 2016-04-12 05:00:00         0
```

```
hourly_steps %>%
  group_by(time) %>%
  summarize(average_steps = mean(StepTotal)) %>%
  ggplot()+
  geom_col(mapping = aes(x=time,y=average_steps, fill=average_steps))+
```

```
  labs(title = "Hourly Steps Throughout the Day", x="",y="")+
  scale_fill_gradient(low = "red", high = "blue")+
  theme(axis.text.x = element_text(angle = 90))
```



In the above graph its clearly visible that the users are active between 8am - 7pm.Mostly active post lunchtime between 12pm-2pm and around evening from 5pm-7pm.

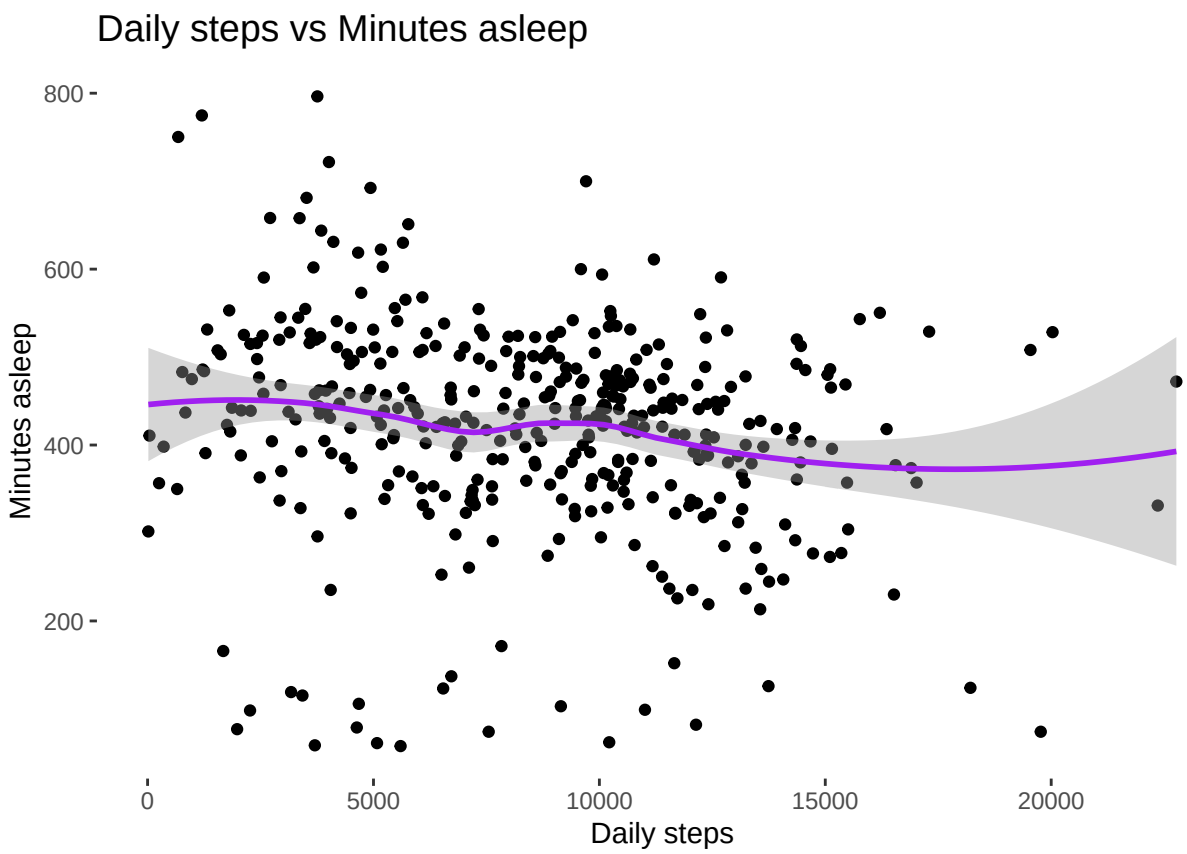
5.3 Searching for Correlations

Lets determine corelations between different variables

. Daily Steps and Sleep . Daily Steps and Calories

```
ggplot(daily_activity_sleep, aes(x = TotalSteps, y = TotalMinutesAsleep))+
  geom_jitter() +
  geom_smooth(color = "purple") +
  labs(title = "Daily steps vs Minutes asleep", x = "Daily steps", y= "Minutes asleep") +
   theme(panel.background = element_blank(),
        plot.title = element_text( size=14))
```

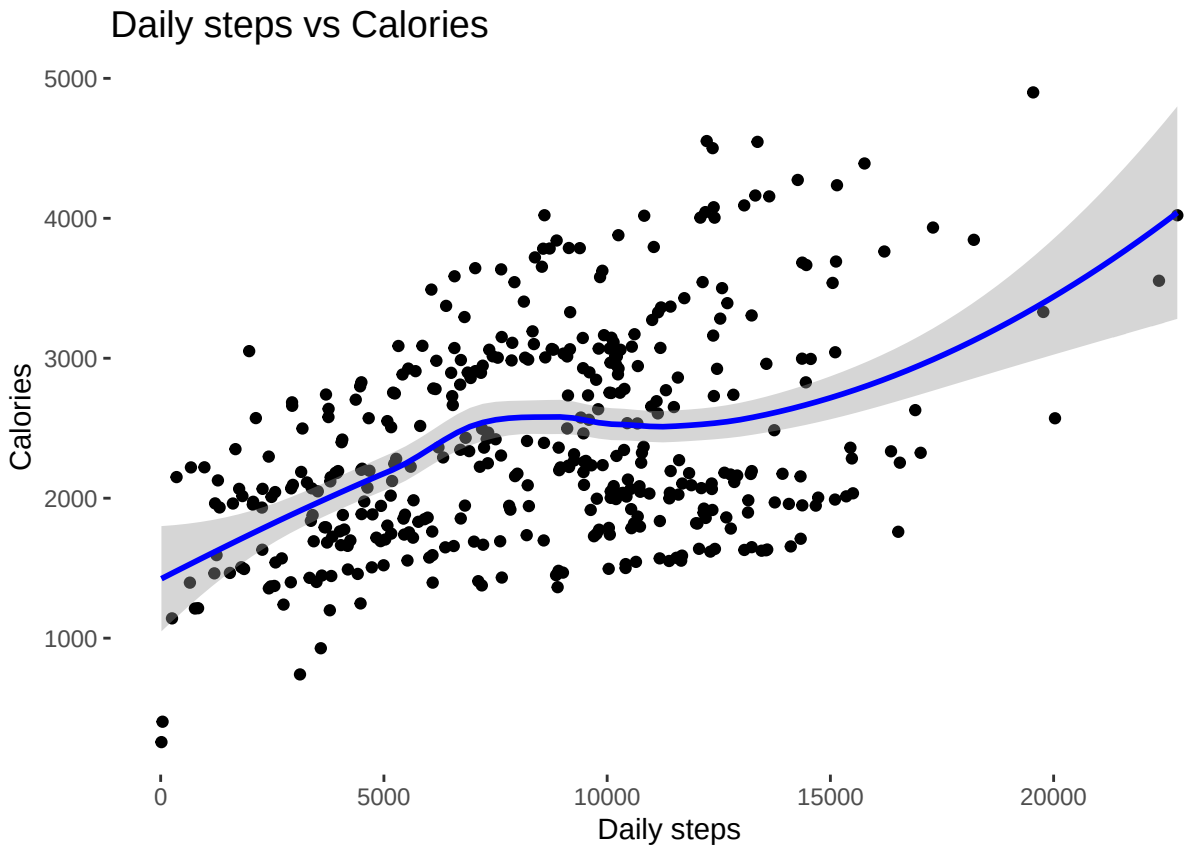
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



As you can see their seems to be no correlation between the steps taken by a user and the amount of sleep in minutes.

```
ggplot(daily_activity_sleep, aes(x=TotalSteps, y=Calories))+
  geom_jitter() +
  geom_smooth(color = "blue") +
  labs(title = "Daily steps vs Calories", x = "Daily steps", y= "Calories") +
   theme(panel.background = element_blank(),
        plot.title = element_text( size=14))
```

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



A positive correlation between steps taken and calories burned is observed here. As assumed the more steps walked the more calories may be burned.

5.4 Usage trend of smart device

We are interested in finding out how frequently the users in our sample use their device now that we have observed certain trends in activity, sleep, and calories burned. By doing so, we can plan our marketing approach and determine which features would make smart gadgets more useful.

Knowing that the date interval is 31 days, we will divide our sample into three categories and determine the proportion of users who use their smart device on a daily basis:

A. Low Use - 1 to 14 days

B. Moderate Use - 15 to 21 days

C. High Use - 22 to 31 days

Creating a data frame for the classification explained above by creating a new column, grouping by Id and calculating number of days used.

```
daily_use <- daily_activity_sleep %>%
  group_by(Id) %>%
  summarize(days_used=sum(n())) %>%
  mutate(usage = case_when(
    days_used >= 1 & days_used <= 14 ~ "Low Use",
    days_used >= 15 & days_used <= 21 ~ "Moderate Use",
    days_used >= 22 & days_used <= 31 ~ "High Use",
  ))
```

```
head(daily_use)
```

```
## # A tibble: 6 x 3
##           Id days_used usage
##        <dbl>     <int> <chr>
## 1 1503960366        25 High Use
## 2 1644430081         4 Low Use
## 3 1844505072         3 Low Use
## 4 1927972279         5 Low Use
## 5 2026352035        28 High Use
## 6 2320127002         1 Low Use
```

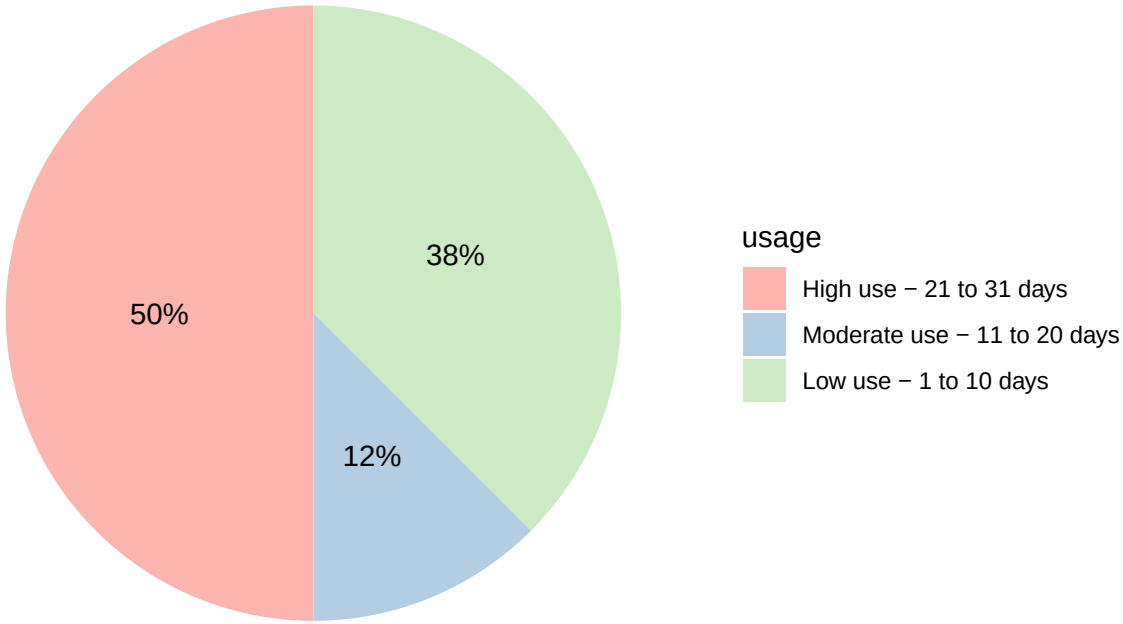
Creating percentage data frame for better visualization in a chart.

```
daily_use_percent <- daily_use %>%
  group_by(usage) %>%
  summarize(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(usage) %>%
  summarize(Total_Percent = total / totals) %>%
  mutate(labels = scales::percent(Total_Percent))

daily_use_percent$usage <- factor(daily_use_percent$usage, levels = c("High Use", "Moderate Use", "Low U

head(daily_use_percent)
```

```
## # A tibble: 3 x 3
##   usage        Total_Percent labels
##   <fct>                <dbl> <chr>
## 1 High Use             0.5   50%
## 2 Low Use              0.375 38%
## 3 Moderate Use         0.125 12%
```

Plot creation with the above result

```
daily_use_percent %>%
  ggplot(aes(x="",y=Total_Percent, fill=usage)) +
  geom_bar(stat = "identity", width = 1)+
  coord_polar("y", start=0)+
  theme_minimal()+
  theme(axis.title.x= element_blank(),
        axis.title.y = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        plot.title = element_text(hjust = 0.5, size=14, face = "bold")) +
  geom_text(aes(label = labels),
            position = position_stack(vjust = 0.5))+
  scale_fill_brewer(palette='Pastel1',
                    labels = c("High use - 21 to 31 days",
                               "Moderate use - 11 to 20 days",
                               "Low use - 1 to 10 days"))+
  labs(title="Daily use of smart device")
```




```
#Creating a DF for a Waffle Chart

datausewaffle = c("High use (16) - 21 to 31 days" = 16, "Moderate use (4) - 11 to 20 days" = 4, "Low us

head(datausewaffle)
```

```
##    High use (16) - 21 to 31 days Moderate use (4) - 11 to 20 days
##                               16                               4
##      Low use (13) - 1 to 10 days
##                               13
```

```
install.packages("waffle")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```
library(waffle)
```

To illustrate the "Usage Type Distribution" in a different visual way, this waffle chart was created. Due to the small sample size, each square represents 1 participant.

```
#WAFFLECHART

waffle(datausewaffle, row = 3, size = 0.33, legend_pos = 'right') +
labs(title = "Usage Group Distribution",
subtitle="1 square = 1 Participant",
caption="Figure 10.1") +
theme(plot.title = element_text(hjust = 0,size = 20, colour = "blue"),
plot.subtitle = element_text(size = 8),
legend.text = element_text(hjust = 0, size = 8),
axis.title = element_text(size = 10),
plot.caption = element_text(size = 7.5))
```

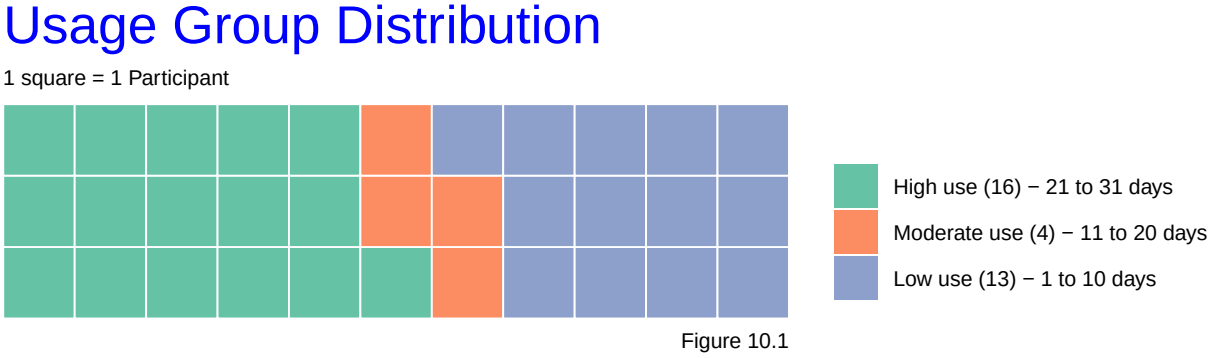


Figure 10.1

Analyzing the above result we can say that

. 50% of the users fall in the category of High usage with 21-31 days. That equates to 16 out of 33 participants.

. 12% of them are moderate users with 11-20 days of usage time.That equates to 4 out of 33 participants.

. 38% users fall in the lowest category with 1-10 days of usage. That makes up 13 of 33 participants

Between 21 and 31 days, a vast majority of users use the device quiet often. This, in my opinion, shows an innate awareness of how the technology is used. That the user must wear it frequently in order to obtain a complete quantitative picture of their activity patterns and physiological state in order to receive the full utility and insights from it.

5.5 Smart Device Usage Timeline/day

As we move further ahead with our analysis, next we will find the total time period for which a user wears the smart device throughout the day and for the same we are merging daily_use and daily_activity data frames.

```
daily_use_merged <- merge(daily_activity, daily_use, by=c ("Id"))
head(daily_use_merged)
```

```
##           Id       date TotalSteps TotalDistance TrackerDistance
## 1 1503960366 2016-05-07      11992          7.71            7.71
## 2 1503960366 2016-05-06      12159          8.03            8.03
## 3 1503960366 2016-05-01      10602          6.81            6.81
## 4 1503960366 2016-04-30      14673          9.25            9.25
## 5 1503960366 2016-04-12      13162          8.50            8.50
## 6 1503960366 2016-04-13      10735          6.97            6.97
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               2.46                     2.12
## 2                        0               1.97                     0.25
## 3                        0               2.29                     1.60
## 4                        0               3.56                     1.42
## 5                        0               1.88                     0.55
## 6                        0               1.57                     0.69
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                3.13                       0                37
## 2                5.81                       0                24
## 3                2.92                       0                33
## 4                4.27                       0                52
## 5                6.06                       0                25
## 6                4.71                       0                21
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories days_used
## 1                  46                  175              833     1821        25
## 2                   6                  289              754     1896        25
```

```
## 3                  35                  246              730      1820         25
## 4                  34                  217              712      1947         25
## 5                  13                  328              728      1985         25
## 6                  19                  217              776      1797         25
##      usage
## 1 High Use
## 2 High Use
## 3 High Use
## 4 High Use
## 5 High Use
## 6 High Use
```

Let's construct a new data frame that divides the total daily minutes consumers spent wearing the gadget into three categories:

1. All Day
2. More than half day
3. Less than half day

```
minutes_worn <- daily_use_merged %>%
  mutate(total_minutes_worn = VeryActiveMinutes +
           FairlyActiveMinutes +
           LightlyActiveMinutes +
           SedentaryMinutes)%>%
  mutate (percent_minutes_worn = (total_minutes_worn/1440)*100) %>%
  mutate (worn = case_when(
    percent_minutes_worn == 100 ~ "All day",
    percent_minutes_worn < 100 & percent_minutes_worn >= 50~ "More than half day",
    percent_minutes_worn < 50 & percent_minutes_worn > 0 ~ "Less than half day"
  ))

head(minutes_worn)
```

```
##           Id       date TotalSteps TotalDistance TrackerDistance
## 1 1503960366 2016-05-07      11992          7.71            7.71
## 2 1503960366 2016-05-06      12159          8.03            8.03
## 3 1503960366 2016-05-01      10602          6.81            6.81
## 4 1503960366 2016-04-30      14673          9.25            9.25
## 5 1503960366 2016-04-12      13162          8.50            8.50
## 6 1503960366 2016-04-13      10735          6.97            6.97
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               2.46                     2.12
## 2                        0               1.97                     0.25
## 3                        0               2.29                     1.60
## 4                        0               3.56                     1.42
## 5                        0               1.88                     0.55
## 6                        0               1.57                     0.69
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                3.13                       0                37
## 2                5.81                       0                24
## 3                2.92                       0                33
## 4                4.27                       0                52
## 5                6.06                       0                25
## 6                4.71                       0                21
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories days_used
```

```
## 1                  46                  175              833     1821        25
## 2                   6                  289              754     1896        25
## 3                  35                  246              730     1820        25
## 4                  34                  217              712     1947        25
## 5                  13                  328              728     1985        25
## 6                  19                  217              776     1797        25
##      usage total_minutes_worn percent_minutes_worn              worn
## 1 High Use               1091             75.76389 More than half day
## 2 High Use               1073             74.51389 More than half day
## 3 High Use               1044             72.50000 More than half day
## 4 High Use               1015             70.48611 More than half day
## 5 High Use               1094             75.97222 More than half day
## 6 High Use               1033             71.73611 More than half day
```

Going ahead we will be creating data frame for better visualization

This data frame will show the total number of users and calculate the total percentage time for which the device was worn.

```
minutes_worn_percent <- minutes_worn%>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(worn) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))

minutes_worn_highuse <- minutes_worn%>%
  filter (usage == "high use")%>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(worn) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))
```

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

```
minutes_worn_moduse <- minutes_worn%>%
  filter(usage == "moderate use") %>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(worn) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))
```

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

```
minutes_worn_lowuse <- minutes_worn%>%
  filter (usage == "low use") %>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(worn) %>%
```

```
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))
```

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

```
minutes_worn_highuse$worn <- factor(minutes_worn_highuse$worn, levels = c("All day", "More than half da
minutes_worn_percent$worn <- factor(minutes_worn_percent$worn, levels = c("All day", "More than half da
minutes_worn_moduse$worn <- factor(minutes_worn_moduse$worn, levels = c("All day", "More than half day"
minutes_worn_lowuse$worn <- factor(minutes_worn_lowuse$worn, levels = c("All day", "More than half day"
```

```
head(minutes_worn_percent)
```

```
## # A tibble: 3 x 3
##   worn               total_percent labels
##   <fct>                      <dbl> <chr>
## 1 All day                   0.365  36%
## 2 Less than half day        0.0351 4%
## 3 More than half day        0.600  60%
```

```
head(minutes_worn_highuse)
```

```
## # A tibble: 0 x 3
## # Groups:   worn [0]
## # ... with 3 variables: worn <fct>, total_percent <dbl>, labels <chr>
```

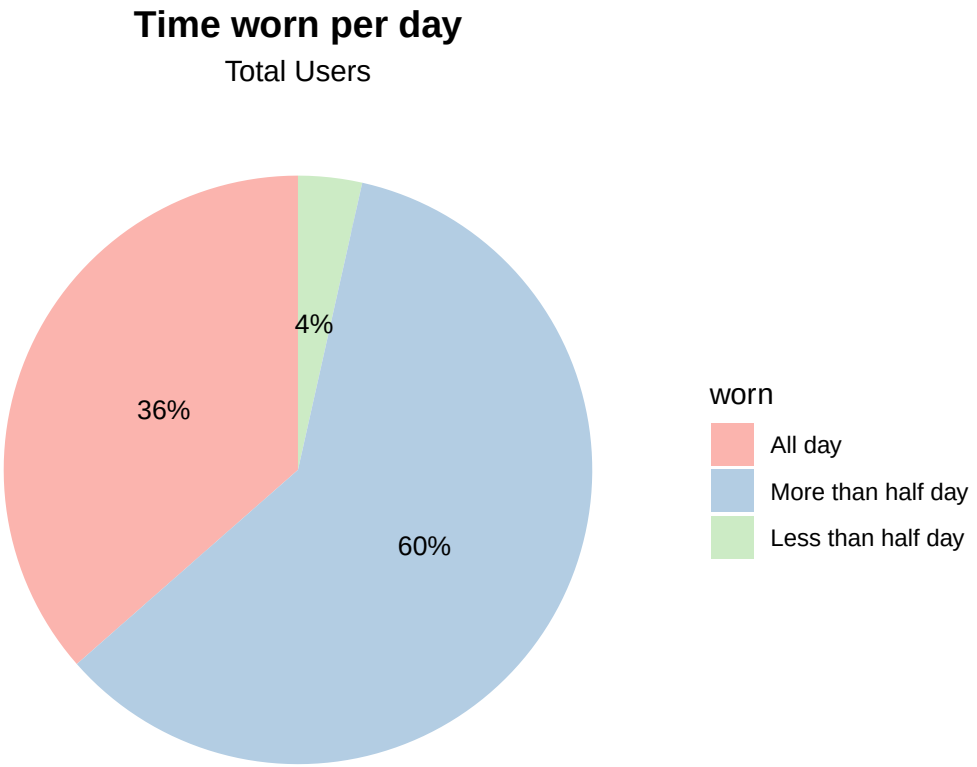
```
head(minutes_worn_moduse)
```

```
## # A tibble: 0 x 3
## # Groups:   worn [0]
## # ... with 3 variables: worn <fct>, total_percent <dbl>, labels <chr>
```

```
head(minutes_worn_lowuse)
```

```
## # A tibble: 0 x 3
## # Groups:   worn [0]
## # ... with 3 variables: worn <fct>, total_percent <dbl>, labels <chr>
```

```
ggplot(minutes_worn_percent, aes(x="",y=total_percent, fill=worn)) +
  geom_bar(stat = "identity", width = 1)+
  coord_polar("y", start=0)+
  theme_minimal()+
  theme(axis.title.x= element_blank(),
        axis.title.y = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        plot.title = element_text(hjust = 0.5, size=14, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5)) +
  scale_fill_brewer(palette='Pastel1')+
  geom_text(aes(label = labels),
            position = position_stack(vjust = 0.5), size = 3.5)+
  labs(title="Time worn per day", subtitle = "Total Users")
```



As per the above chart we can see that 36% of the total of users wear the device all day long, 60% more than half day long and just 4% for less than half a day.

6. ACT Phase (Conclusion)

Bellabeat's goal is to empower women by giving them the knowledge to learn about themselves.

Based on our findings, I would advise using your own monitoring data for more analysis in order for us to respond to our business task and assist Bellabeat with their purpose. Since we lacked information on the users' demographics, the datasets we used have a limited sample and are potentially biased. Knowing that young and adult women are our primary target demographic, I would encourage you to keep looking for trends so you can develop a marketing strategy centred on them.

Having said that, following our investigation, we identified a number of tendencies that could benefit our online campaign and enhance the Bellabeat app:

1. Daily app notifications for actions and posts: We divided users into 4 categories and discovered that, aside from Sundays, each user takes an average of more than 7,500 steps per day. By giving them alarms if they haven't reached the CDC's suggested daily step total of 8,000 and by making posts on our app outlining the advantages of achieving that target, we can push users to do so. According to the CDC, the death rate decreases the more steps you take. Additionally, we observed a favourable connection between steps and calories.
2. Techniques for notification and sleep: Our findings indicate that users sleep for fewer than eight hours per day. They may program a desired bedtime and get a notification a few minutes beforehand to get ready for bed. Provide your customers with more sleep-related resources, such as breathing tips, music podcasts, and sleep strategies.
3. Reward systems: Because we are aware that some users are not motivated by notifications, we considered developing a brief game for our app. Reaching different stages in the game depends on how many steps you take each day. For a while (perhaps a month), you must keep up your activity level in order to

advance to the next level. You receive a set number of stars for completing each level, which may be redeemed for gifts or discounts on other Bellabeat products.