Bellabeat Smart Device

Ivan Fung

2022-10-30

Company

Bellabeat is a small successful high-tech manufacturer of health-focused products for women, and it offers different smart devices that collect data on activity, sleep, stress, and reproductive health to empower women with knowledge about their own health and habits. Management of Bellabeat see the potential growth of the company to become a major player in the global smart device market. In 2016 Bellabeat has launched multiple products and expanded their business globally. Initially the products became available on their own e-commerce channel and online retailers. Bellabeat has been focused on digital marketing extensively, including Google Search, video ads, and consumer engagement on social media platforms.

The Business Task

As a data analyst working on the marketing analyst team, I was asked to analyze the available open smart device data to gain insight into consumers usage of their smart devices. As part of my tasks, I have to apply my insights on one of the Bellabeat's exsiting products for the future marketing campaign.

Existing Products

Bellabeat app: provides users with health data related data to their activity, sleep, stress, menstrual cycle, etc.

Leaf: classic wellness tracker can be worn as a bracelet, necklace, or clip.

Time: wellness watch combines the timeless look of a classic timepiece with smart technology

Spring: a water bottle that tracks daily water intake using smart technology

- The focus will be on Bellabeat App for recommendations.

Key Stakeholders

- Urška Sršen: cofounder and Chief Creative Officer.
- Sando Mur: cofounder and a key member of the Bellabeat executive team.
- Bellabeat marketing analytics team

Prepare

There are 18 CSV files available from FitBit Fitness Tracker Data (CC0: Public Domain, dataset made available through Mobius). The datasets were generated by respondents to a distributed survey via Amazon Mechanical Turk, the dataset has been verified to be open-source. As a result, it can be copied, modified, and distributed without permission.

Datasets includes 33 Fitbit user activity and data over a month period (31 days) that will help answer the business tasks.

R and RStudio were adopted to prepare, process, analyze and visualize the datasets. The files will be downloaded to local PC and was carrying out within the Rstudio.

Data limitations

- The data provided is outdated (2016).
- Small sample size (8-33 users within various data sets).
- Demographic information, such as gender and age is unavailable.
- Sampling bias may be present

Loading the packages

```
library(ggplot2)       # Visualization
library(ggthemes)      # Visualization
library(scales)        # Visualization
library(dplyr)         # Data Manipulation

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(mice)          # Imputation

##
## Attaching package: 'mice'

## The following object is masked from 'package:stats':
##
##     filter

## The following objects are masked from 'package:base':
##
##     cbind, rbind

library(tidyverse)
```

```
## ── Attaching packages
## ───────────────────────────────────────
## tidyverse 1.3.2 ──

## ✓ tibble  3.1.7     ✓ purrr   0.3.4
## ✓ tidyr   1.2.0     ✓ stringr 1.4.0
## ✓ readr   2.1.2     ✓ forcats 0.5.1
## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✕ readr::col_factor() masks scales::col_factor()
## ✕ purrr::discard()    masks scales::discard()
## ✕ mice::filter()      masks dplyr::filter(), stats::filter()
## ✕ dplyr::lag()        masks stats::lag()

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(tidyr)
```

Among the 18 csv files which has been provided from the datasets, I only selected few relevant data files for the analysis. These files included:

- dailyActivity_merged
- sleepDay_merged
- hourlySteps_merged

From the preliminary review of these files, most of the files are incomplete with limited participants' data recorded. As a result, I chose files which have the most recorded data of most of the participants (33 participants) which were collected by the providers during that time.

Importing the datasets

```
Daily_Activity.data <- read.csv("C:/Users/ikinf/OneDrive/Desktop/Ivan/Google
Analytics/Case studies/Datasets/dailyActivity_merged.csv")

Sleep_Data<- read.csv("C:/Users/ikinf/OneDrive/Desktop/Ivan/Google
Analytics/Case studies/Datasets/sleepDay_merged.csv")

hourly_steps <- read.csv("C:/Users/ikinf/OneDrive/Desktop/Ivan/Google
Analytics/Case studies/Datasets/hourlySteps_merged.csv")
```

Preview of the imported datasets

By using the function View() and glimpse() to get an overview of the datasets and the summary of each column

```
View(Daily_Activity.data)
glimpse(Daily_Activity.data)

## Rows: 940
## Columns: 15
## $ Id                       <dbl> 1503960366, 1503960366, 1503960366,
150396036…
## $ ActivityDate             <chr> "4/12/2016", "4/13/2016", "4/14/2016",
"4/15/…
## $ TotalSteps               <int> 13162, 10735, 10460, 9762, 12669, 9705,
13019…
## $ TotalDistance            <dbl> 8.50, 6.97, 6.74, 6.28, 8.16, 6.48, 8.59,
9.8…
## $ TrackerDistance          <dbl> 8.50, 6.97, 6.74, 6.28, 8.16, 6.48, 8.59,
9.8…
## $ LoggedActivitiesDistance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, …
## $ VeryActiveDistance       <dbl> 1.88, 1.57, 2.44, 2.14, 2.71, 3.19, 3.25,
3.5…
## $ ModeratelyActiveDistance <dbl> 0.55, 0.69, 0.40, 1.26, 0.41, 0.78, 0.64,
1.3…
## $ LightActiveDistance      <dbl> 6.06, 4.71, 3.91, 2.83, 5.04, 2.51, 4.71,
5.0…
## $ SedentaryActiveDistance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, …
## $ VeryActiveMinutes        <int> 25, 21, 30, 29, 36, 38, 42, 50, 28, 19,
66, 4…
## $ FairlyActiveMinutes      <int> 13, 19, 11, 34, 10, 20, 16, 31, 12, 8,
27, 21…
## $ LightlyActiveMinutes     <int> 328, 217, 181, 209, 221, 164, 233, 264,
205, …
## $ SedentaryMinutes         <int> 728, 776, 1218, 726, 773, 539, 1149, 775,
818…
## $ Calories                 <int> 1985, 1797, 1776, 1745, 1863, 1728, 1921,
203…
```

```
View(Sleep_Data)
glimpse(Sleep_Data)

## Rows: 413
## Columns: 5
## $ Id                 <dbl> 1503960366, 1503960366, 1503960366, 1503960366,
150…
## $ SleepDay           <chr> "4/12/2016", "4/13/2016", "4/15/2016",
"4/16/2016",…
## $ TotalSleepRecords  <int> 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, …
## $ TotalMinutesAsleep <int> 327, 384, 412, 340, 700, 304, 360, 325, 361,
430, 2…
## $ TotalTimeInBed     <int> 346, 407, 442, 367, 712, 320, 377, 364, 384,
449, 3…

View(hourly_steps)
glimpse(hourly_steps)

## Rows: 22,099
## Columns: 3
## $ Id           <dbl> 1503960366, 1503960366, 1503960366, 1503960366,
150396036…
## $ ActivityHour <chr> "4/12/2016 0:00", "4/12/2016 1:00", "4/12/2016 2:00",
"4/…
## $ StepTotal    <int> 373, 160, 151, 0, 0, 0, 0, 0, 250, 1864, 676, 360,
253, 2…
```

Process

Cleaning and formatting After reviewing the data strucutures, we would process the data cleaning and any necessary formatting before proceeding to the Analyze phase of the case study.

Identify Duplicates data in dataset

```
sum(duplicated(Daily_Activity.data))

## [1] 0

sum(duplicated(Sleep_Data))

## [1] 3

sum(duplicated(hourly_steps))

## [1] 0
```

There are 3 duplicates in Sleep Data dataframe; we will drop the duplicates before proceeding #Use unique function to return only the uniques values for the dataframe

```
Sleep_Data <- unique(Sleep_Data)
```

#To verify if there is any more duplicates remain in the dataframe; we will use the sum(duplicated()) functions

```
sum(duplicated(Sleep_Data))

## [1] 0
```

Converting Date format from character to date format in dataframes: column "SleepDate","ActivityDate" & "ActivityHour"

After reviewing the data "Date" columns in each dataset, I noticed that they are all in character type. As a result, I reformatted these columns into Date or Date_time format for our further analysis.
I used as.Date() and as.POSIXct() for formatting the columns into Date or Date time format.

```
Daily_Activity.data$ActivityDate <- as.Date(Daily_Activity.data$ActivityDate,
format= "%m/%d/%Y")

View(Daily_Activity.data)

str(Daily_Activity.data)

## 'data.frame':    940 obs. of  15 variables:
##  $ Id                      : num  1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09
...
##  $ ActivityDate            : Date, format: "2016-04-12" "2016-04-13" ...
##  $ TotalSteps              : int  13162 10735 10460 9762 12669 9705 13019
15506 10544 9819 ...
##  $ TotalDistance           : num  8.5 6.97 6.74 6.28 8.16 ...
##  $ TrackerDistance         : num  8.5 6.97 6.74 6.28 8.16 ...
##  $ LoggedActivitiesDistance: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveDistance      : num  1.88 1.57 2.44 2.14 2.71 ...
##  $ ModeratelyActiveDistance: num  0.55 0.69 0.4 1.26 0.41 ...
##  $ LightActiveDistance     : num  6.06 4.71 3.91 2.83 5.04 ...
##  $ SedentaryActiveDistance : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveMinutes       : int  25 21 30 29 36 38 42 50 28 19 ...
##  $ FairlyActiveMinutes     : int  13 19 11 34 10 20 16 31 12 8 ...
##  $ LightlyActiveMinutes    : int  328 217 181 209 221 164 233 264 205 211
...
##  $ SedentaryMinutes        : int  728 776 1218 726 773 539 1149 775 818
838 ...
##  $ Calories                : int  1985 1797 1776 1745 1863 1728 1921 2035
1786 1775 ...

Sleep_Data$SleepDay <- as.Date(Sleep_Data$SleepDay, format= "%m/%d/%Y")

View(Sleep_Data)

str(Sleep_Data)
```

```
## 'data.frame':    410 obs. of  5 variables:
##  $ Id                : num  1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ SleepDay          : Date, format: "2016-04-12" "2016-04-13" ...
##  $ TotalSleepRecords : int  1 2 1 2 1 1 1 1 1 1 ...
##  $ TotalMinutesAsleep: int  327 384 412 340 700 304 360 325 361 430 ...
##  $ TotalTimeInBed    : int  346 407 442 367 712 320 377 364 384 449 ...
```

```
hourly_steps$date_time <-as.POSIXct(hourly_steps$ActivityHour, format =
"%m/%d/%Y %H:%M")

View(hourly_steps)

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

head(hourly_steps)
```

```
##           Id  ActivityHour StepTotal       date     time
## 1 1503960366 4/12/2016 0:00       373 2016-04-12 00:00:00
## 2 1503960366 4/12/2016 1:00       160 2016-04-12 01:00:00
## 3 1503960366 4/12/2016 2:00       151 2016-04-12 02:00:00
## 4 1503960366 4/12/2016 3:00         0 2016-04-12 03:00:00
## 5 1503960366 4/12/2016 4:00         0 2016-04-12 04:00:00
## 6 1503960366 4/12/2016 5:00         0 2016-04-12 05:00:00
```

Rename column names in datafrane Daily_Activiy.data and Sleep_Data

```
Daily <- rename(Daily_Activity.data, Date = ActivityDate)

Sleep <- rename(Sleep_Data, Date = SleepDay)

head(Daily)
```

```
##           Id       Date TotalSteps TotalDistance TrackerDistance
## 1 1503960366 2016-04-12      13162          8.50            8.50
## 2 1503960366 2016-04-13      10735          6.97            6.97
## 3 1503960366 2016-04-14      10460          6.74            6.74
## 4 1503960366 2016-04-15       9762          6.28            6.28
## 5 1503960366 2016-04-16      12669          8.16            8.16
## 6 1503960366 2016-04-17       9705          6.48            6.48
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               1.88                     0.55
## 2                        0               1.57                     0.69
## 3                        0               2.44                     0.40
## 4                        0               2.14                     1.26
## 5                        0               2.71                     0.41
## 6                        0               3.19                     0.78
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                6.06                       0                25
## 2                4.71                       0                21
## 3                3.91                       0                30
## 4                2.83                       0                29
## 5                5.04                       0                36
## 6                2.51                       0                38
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories
## 1                  13                  328              728     1985
## 2                  19                  217              776     1797
## 3                  11                  181             1218     1776
## 4                  34                  209              726     1745
## 5                  10                  221              773     1863
## 6                  20                  164              539     1728
```

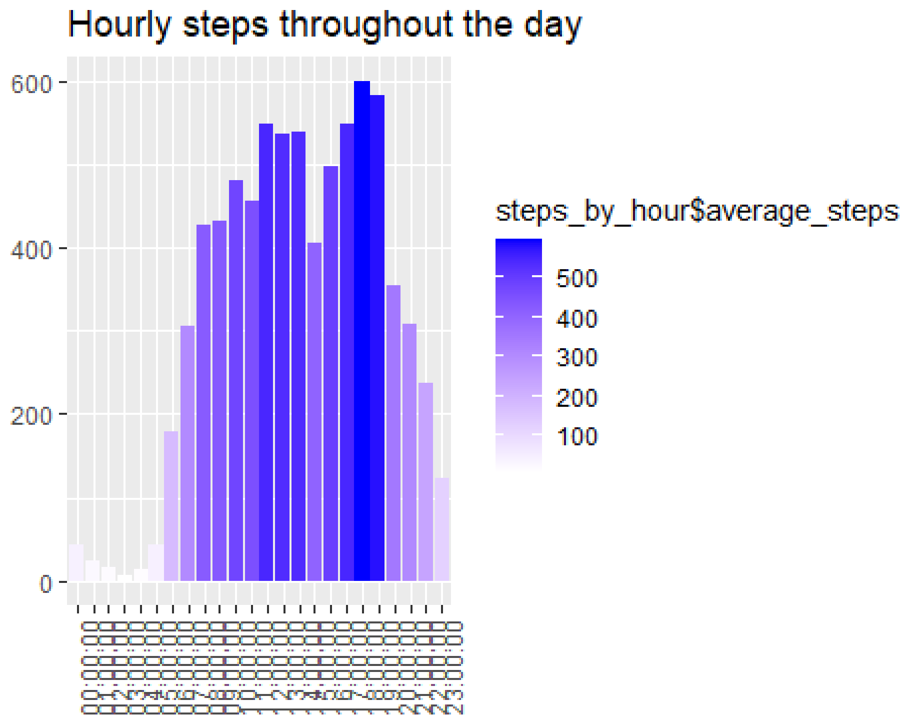
```
head(Sleep)
```

```
##           Id       Date TotalSleepRecords TotalMinutesAsleep
TotalTimeInBed
## 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
```

Analyze Phase and Share Phase

In my first part of the analysis, I used the hourly_step dataset to gain insight about the peak hours which the users will have the most activities (number of steps) of each day. As a result, I can see the different level of hourly steps throughout the day by appling groupby() and summarize() functions to capture the average steps in total from each hour.

```
steps_by_hour <- hourly_steps%>%
  group_by(time) %>%
  summarize(average_steps = mean(StepTotal))

View(steps_by_hour)



ggplot() +
  geom_col(mapping = aes(x=steps_by_hour$time, y =
steps_by_hour$average_steps, fill = steps_by_hour$average_steps)) +
  labs(title = "Hourly steps throughout the day", x="", y="") +
  scale_fill_gradient(low = "white", high = "blue")+
  theme(axis.text.x = element_text(angle = 90))
```



As illustrated in the visualization of bar charts, users were more active during the day time from 8 am to 8 p.m., and the peak hours with most average steps taken by the user are between lunch hours (12 am to 2 pm) and evening from 5 pm to 7 pm.

Correlation among different variables

After analysis the hourly steps taken by the users, I will proceed with the analysis of the correlation amount different variables. I selected three variables which I believed would have direct relationship with the Calories burned recored by the smart device. Since I want to look at the correlations among the number of TotalSteps, TotalDistance, Calories burned, and amount of Sedentary minutes, I have to merged two dataframes: Daily_Activity.data and Sleep_Data for analysis. I used the merge function() to combine the tables together, and use the “id” and “Date” as the keys for matching the tables together.

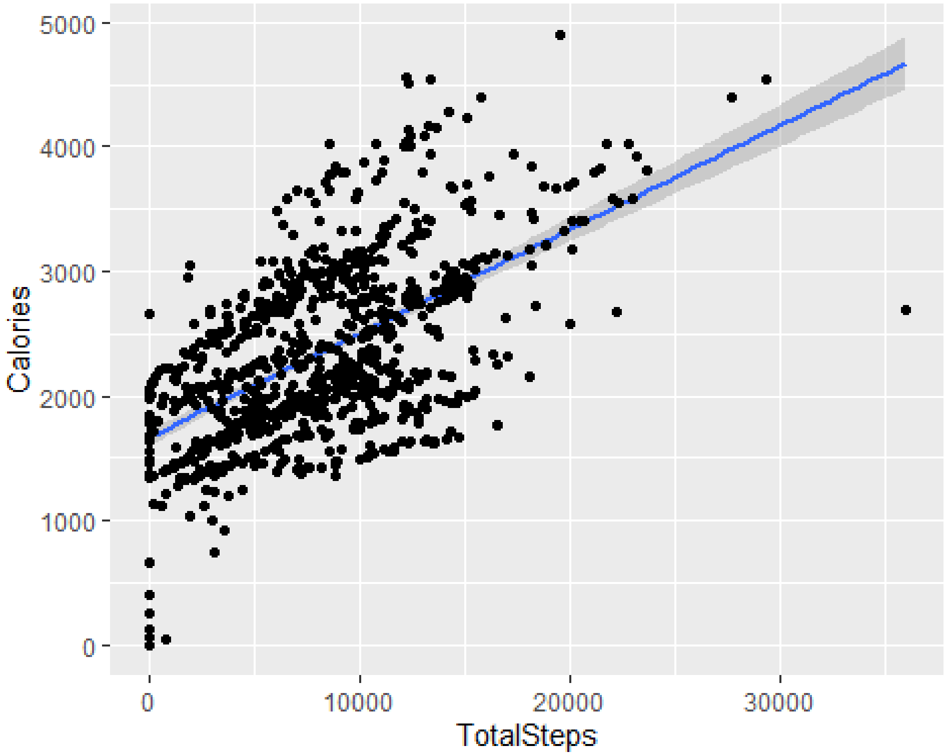
```
data_df <- merge(Daily, Sleep, By=c("id", "Date"), all.x = TRUE)

View(data_df)


glimpse (data_df)

## Rows: 940
## Columns: 18
## $ Id                          <dbl> 1503960366, 1503960366, 1503960366,
150396036…
## $ Date                     <date> 2016-04-12, 2016-04-13, 2016-04-14,
2016-04-…
## $ TotalSteps               <int> 13162, 10735, 10460, 9762, 12669, 9705,
13019…
## $ TotalDistance            <dbl> 8.50, 6.97, 6.74, 6.28, 8.16, 6.48, 8.59,
9.8…
## $ TrackerDistance          <dbl> 8.50, 6.97, 6.74, 6.28, 8.16, 6.48, 8.59,
9.8…
## $ LoggedActivitiesDistance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, …
## $ VeryActiveDistance       <dbl> 1.88, 1.57, 2.44, 2.14, 2.71, 3.19, 3.25,
3.5…
## $ ModeratelyActiveDistance <dbl> 0.55, 0.69, 0.40, 1.26, 0.41, 0.78, 0.64,
1.3…
## $ LightActiveDistance      <dbl> 6.06, 4.71, 3.91, 2.83, 5.04, 2.51, 4.71,
5.0…
## $ SedentaryActiveDistance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, …
## $ VeryActiveMinutes        <int> 25, 21, 30, 29, 36, 38, 42, 50, 28, 19,
66, 4…
## $ FairlyActiveMinutes      <int> 13, 19, 11, 34, 10, 20, 16, 31, 12, 8,
27, 21…
## $ LightlyActiveMinutes     <int> 328, 217, 181, 209, 221, 164, 233, 264,
205, …
## $ SedentaryMinutes         <int> 728, 776, 1218, 726, 773, 539, 1149, 775,
818…
## $ Calories                 <int> 1985, 1797, 1776, 1745, 1863, 1728, 1921,
203…
## $ TotalSleepRecords        <int> 1, 2, NA, 1, 2, 1, NA, 1, 1, 1, NA, 1, 1,
1, …
## $ TotalMinutesAsleep       <int> 327, 384, NA, 412, 340, 700, NA, 304,
360, 32…
## $ TotalTimeInBed           <int> 346, 407, NA, 442, 367, 712, NA, 320,
377, 36…
```

```
str(data_df)
```

```
## 'data.frame':    940 obs. of  18 variables:
##  $ Id                      : num  1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09
...
##  $ Date                    : Date, format: "2016-04-12" "2016-04-13" ...
##  $ TotalSteps              : int  13162 10735 10460 9762 12669 9705 13019
15506 10544 9819 ...
##  $ TotalDistance           : num  8.5 6.97 6.74 6.28 8.16 ...
##  $ TrackerDistance         : num  8.5 6.97 6.74 6.28 8.16 ...
##  $ LoggedActivitiesDistance: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveDistance      : num  1.88 1.57 2.44 2.14 2.71 ...
##  $ ModeratelyActiveDistance: num  0.55 0.69 0.4 1.26 0.41 ...
##  $ LightActiveDistance     : num  6.06 4.71 3.91 2.83 5.04 ...
##  $ SedentaryActiveDistance : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveMinutes       : int  25 21 30 29 36 38 42 50 28 19 ...
##  $ FairlyActiveMinutes     : int  13 19 11 34 10 20 16 31 12 8 ...
##  $ LightlyActiveMinutes    : int  328 217 181 209 221 164 233 264 205 211
...
##  $ SedentaryMinutes        : int  728 776 1218 726 773 539 1149 775 818
838 ...
##  $ Calories                : int  1985 1797 1776 1745 1863 1728 1921 2035
1786 1775 ...
##  $ TotalSleepRecords       : int  1 2 NA 1 2 1 NA 1 1 1 ...
##  $ TotalMinutesAsleep      : int  327 384 NA 412 340 700 NA 304 360 325
...
##  $ TotalTimeInBed          : int  346 407 NA 442 367 712 NA 320 377 364
...
```

Correlation between variables: TotalSteps & Calories

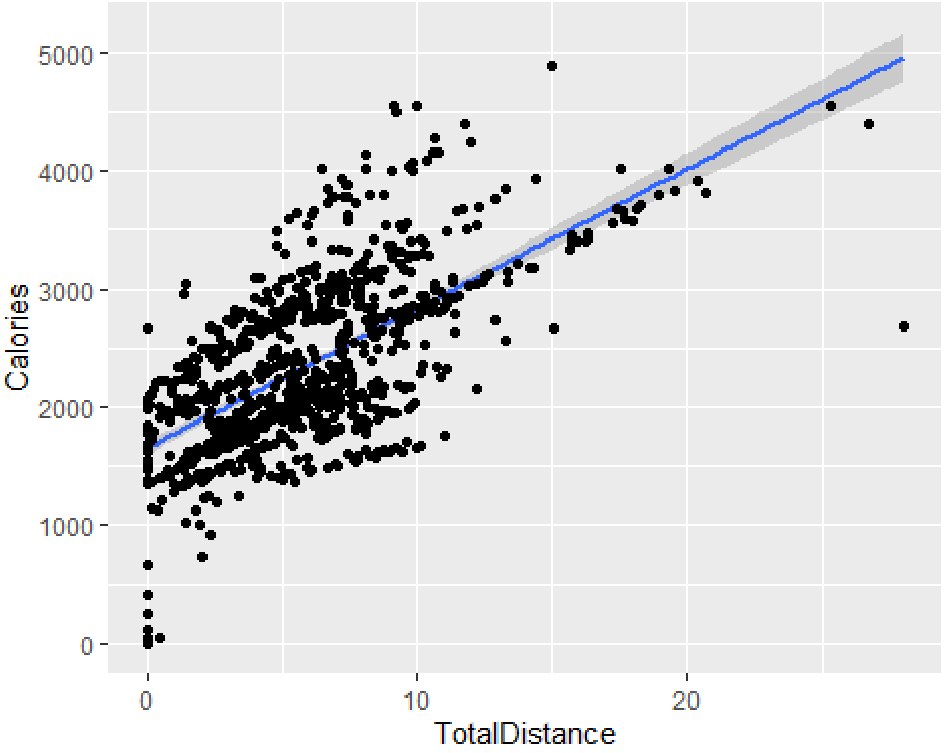
```
ggplot(data=data_df) + geom_smooth(mapping = aes(x=TotalSteps, y=Calories),
method ="lm") +
  geom_point(mapping = aes(x=TotalSteps, y=Calories))

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



Scatter plot between Calories and Total Steps shown there was a positive correlation between these two variables. It makes sense that users can have more calories burned with more steps taken by the users.

Correlation between variables: TotalDistance & Calories

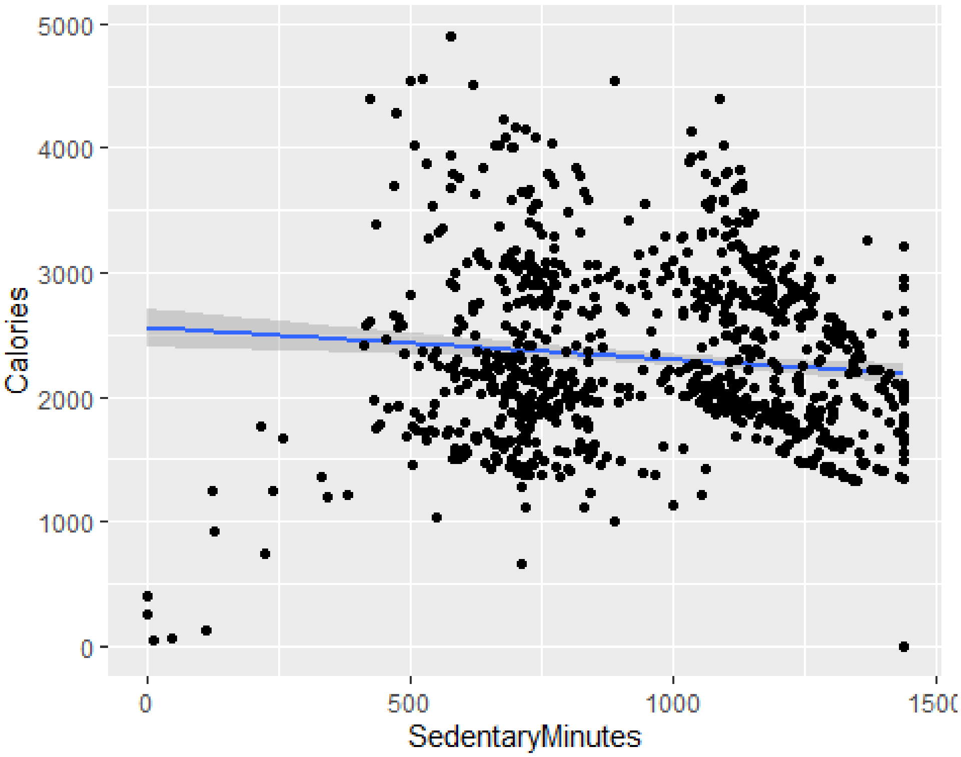
```
ggplot(data=data_df) + geom_smooth(mapping = aes(x=TotalDistance,
y=Calories), method ="lm") +
  geom_point(mapping = aes(x=TotalDistance, y=Calories))

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



Similarly, scatter plot between Calories and TotalDistance shown there was a positive correlation between these two variables. More calories will be burned with more walking distance being taken by the users.

Correlation between variables: SedentaryMinutes & Calories

```
ggplot(data=data_df) + geom_smooth(mapping = aes(x=SedentaryMinutes,
y=Calories), method ="lm") +
  geom_point(mapping = aes(x=SedentaryMinutes, y=Calories))

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



Scatter plot between Calories and Sedentary Minutes shown there was a slight negative correlation between these two variables. It reveals that the more sedentary minutes the users took, the less calories burned would be recorded by the smart device.

Use of smart device

After finding some relationships about the calories burned with other variables. I decided to look into data relating to Days used smart device by the users to have better understanding of the usage of smart device in users' daily activities. It is important to know how often do the users in the sample population use their device, so that we can plan our marketing strategy and see what features would benefit the use of smart devices.

I classified the sample data into three categories in accordance with the amount of time they used the smart device during the observation period.

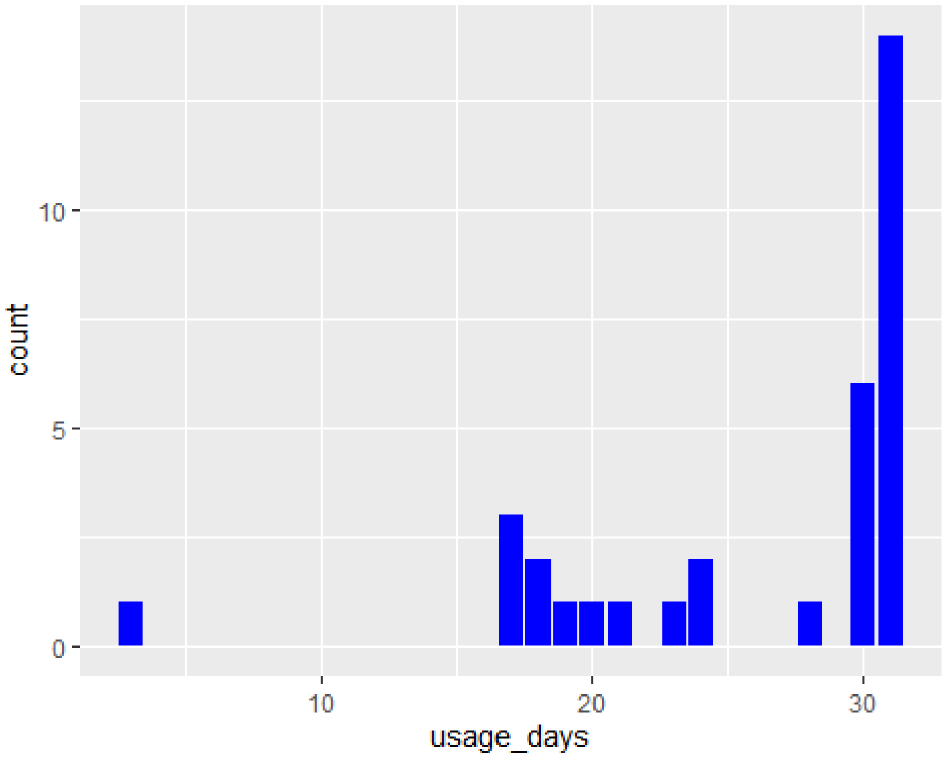
- high usage - wearing the device 80% of the whole monitoring period
- moderate usage - wearing the device 40-80% of the whole monitoring period
- low usage - wearing the devices less 40% of the whole monitoring period

I created a new dataframe which will group by all users by id to determine the percentage of day time which the users have readings from the device within the monitoring period. A new column will be created so that we classify whether user was a high-usage, moderate-usage, low-usage participant

```
usage_data <- data_df %>%
  filter(TotalSteps !=0) %>%
  group_by(Id) %>%
  summarize(usage_days =(sum(n())))

usage_data <- usage_data %>%
  mutate(percentage_usage = ((usage_days/31)*100))

usage_data$percentage_usage <- round(usage_data$percentage_usage, digits =2)

View(usage_data)

str(usage_data)

## tibble [33 × 3] (S3: tbl_df/tbl/data.frame)
##  $ Id              : num [1:33] 1.50e+09 1.62e+09 1.64e+09 1.84e+09
1.93e+09 ...
##  $ usage_days      : int [1:33] 30 31 30 21 17 31 31 31 18 31 ...
##  $ percentage_usage: num [1:33] 96.8 100 96.8 67.7 54.8 ...

usage_data$Usage_Level[usage_data$percentage_usage <=40] <- "Low"

## Warning: Unknown or uninitialised column: `Usage_Level`.

usage_data$Usage_Level[usage_data$percentage_usage >40 &
usage_data$percentage_usage<=80] <- "Moderate"

usage_data$Usage_Level[usage_data$percentage_usage >80 &
usage_data$percentage_usage<=100 ] <- "High"


View(usage_data)
```

Using the ggplot, the usage level was plotted in bar chart to demonstrate the frequency of each usage level

```
ggplot(data = usage_data, aes(x = usage_days)) +
  geom_bar(stat = "count", position = "dodge", fill = "Blue")
```



```
mean_time_usage <- usage_data %>%
  summarise(mean(usage_days))

mean_time_usage

## # A tibble: 1 × 1
##   `mean(usage_days)`
##                <dbl>
## 1               26.2
```

According to the bar charts, the majority of the users wore the device for at least 80% of the time. The average time wore by the whole sample population was 26.15 days.

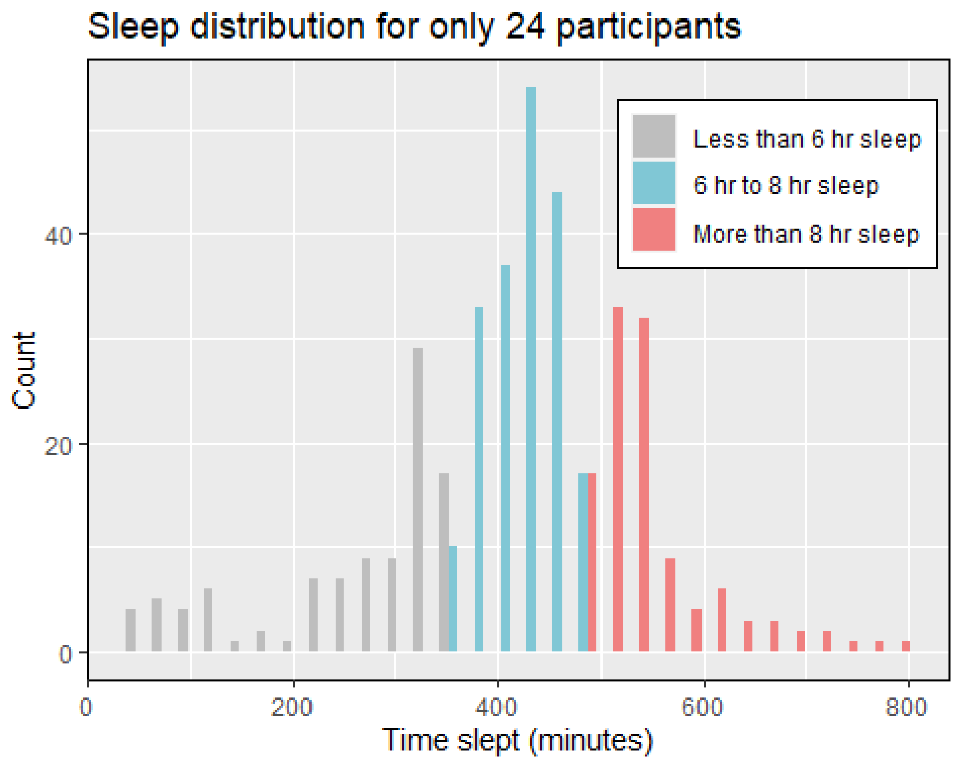
Based on the Sleep_Data file, the average sleeping time for each user in the sample population was determined. Using the group_by() and summarise() functions to get insight about the average number of hours of each of the 24 participants. However, there were some missing number in the dataset, and there are only participants whom had the sleeping records. I would still continue the analysis to get an idea about the sleeping time for these 24 participants even there were less than the minimal sample size of 30.

```
 average_sleep <- Sleep_Data %>%
  group_by(Id) %>%
  summarise (mean_sleep_time = mean(TotalMinutesAsleep))


summary(average_sleep)
```

```
##        Id              mean_sleep_time
##  Min.   :1.504e+09   Min.   : 61.0
##  1st Qu.:2.340e+09   1st Qu.:336.3
##  Median :4.502e+09   Median :417.2
##  Mean   :4.764e+09   Mean   :377.4
##  3rd Qu.:6.822e+09   3rd Qu.:449.3
##  Max.   :8.792e+09   Max.   :652.0

#mean_avg_sleep <- average_sleep %>%
#  summarise(mean(mean_sleep_time))

#mean_avg_sleep
```

The average sleeping time from the 24 participants over the observation period was about 377.4 minutes (about 6.3 hours). The maximum number of sleeping time was 652 minutes while the minimum sleep time recorded was 61 minutes.

```
Sleep_Data <- Sleep_Data %>%
  mutate(sleep_quality = ifelse(TotalMinutesAsleep <= 360, 'Less than 6 hr
sleep',
                         ifelse(TotalMinutesAsleep <= 480, '6 hr to 8 hr
sleep',
                         'More than 8 hr sleep'))) %>%
  mutate(sleep_quality = factor(sleep_quality,
                          levels = c('Less than 6 hr sleep','6 hr to 8 hr
sleep',
                                      'More than 8 hr sleep')))

View(Sleep_Data)

ggplot(Sleep_Data, aes(x = TotalMinutesAsleep, fill = sleep_quality)) +
  geom_histogram(position = 'dodge', bins = 30) +
  scale_fill_manual(values=c("grey", "#80c7d5", "lightcoral")) +
  theme(legend.position = c(.80, .80),
        legend.title = element_blank(),
        legend.spacing.y = unit(0, "mm"),
        panel.border = element_rect(colour = "black", fill=NA),
        legend.background = element_blank(),
        legend.box.background = element_rect(colour = "black")) +
  labs(
    title = "Sleep distribution for only 24 participants",
    x = "Time slept (minutes)",
    y = "Count")
```



After look at the sleep time of the users, I studied the daily activity dataframe and to determine the average daily activity, average Calories burned, and the average sedentary time for each user.
We can determine the type of users in accordance with the average daily steps she took each day. The classification has been made from the article !https://www.10000steps.org.au/articles/counting-steps/ and the CDC Daily Steps Recommendation !https://www.cdc.gov/media/releases/2020/p0324-daily-step-count.html

- Sedentary - Less than 5000 steps a day.
- Lightly active - Between 5000 and 7499 steps a day.
- Fairly active - Between 7500 and 9999 steps a day.
- Very active - More than 10000 steps a day.

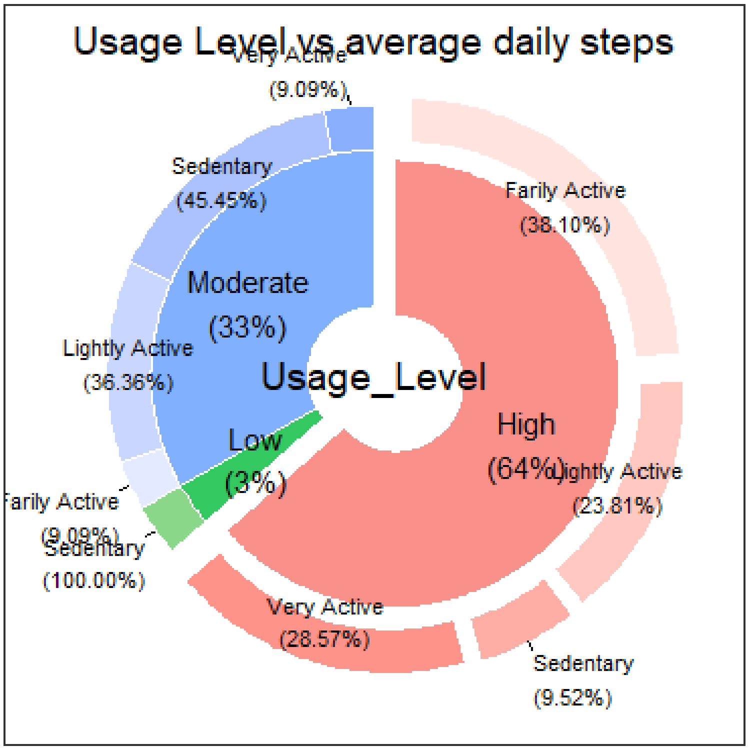
```
daily_average <- data_df %>%
  group_by(Id) %>%
  summarise (mean_daily_steps = mean(TotalSteps), mean_daily_calories =
mean(Calories), mean_sedentarytime = mean(SedentaryMinutes))

View(daily_average)

daily_average <- daily_average %>%
  mutate(steps_category = 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 ~ "Farily Active",
         mean_daily_steps >=10000 ~ "Very Active"))
View(daily_average)
```

To get insight of the particular usage-level of the smart device user would lead to more exercise and sedentary time, I merged 2 subsets of the dataframes: usage_data and daily_average

```
usage_mean_activity <- merge(daily_average, average_sleep, By="Id", all =
TRUE)

View(usage_mean_activity)

usage_level_mean <- merge(usage_data, usage_mean_activity, By="Id", all =
TRUE)

View(usage_level_mean)
```

To illustrate the results, I constructed the pie chart for the visualization.

```
library(webr)

PieDonut(usage_level_mean, aes(Usage_Level, steps_category),
         explode = 1,
         explodeDonut = TRUE,
         title = "Usage Level vs average daily steps")

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use
`guides(<scale> =
## "none")` instead.

## Warning: Ignoring unknown aesthetics: explode

## Warning: `guides(<scale> = FALSE)` is deprecated. Please use
`guides(<scale> =
## "none")` instead.
```

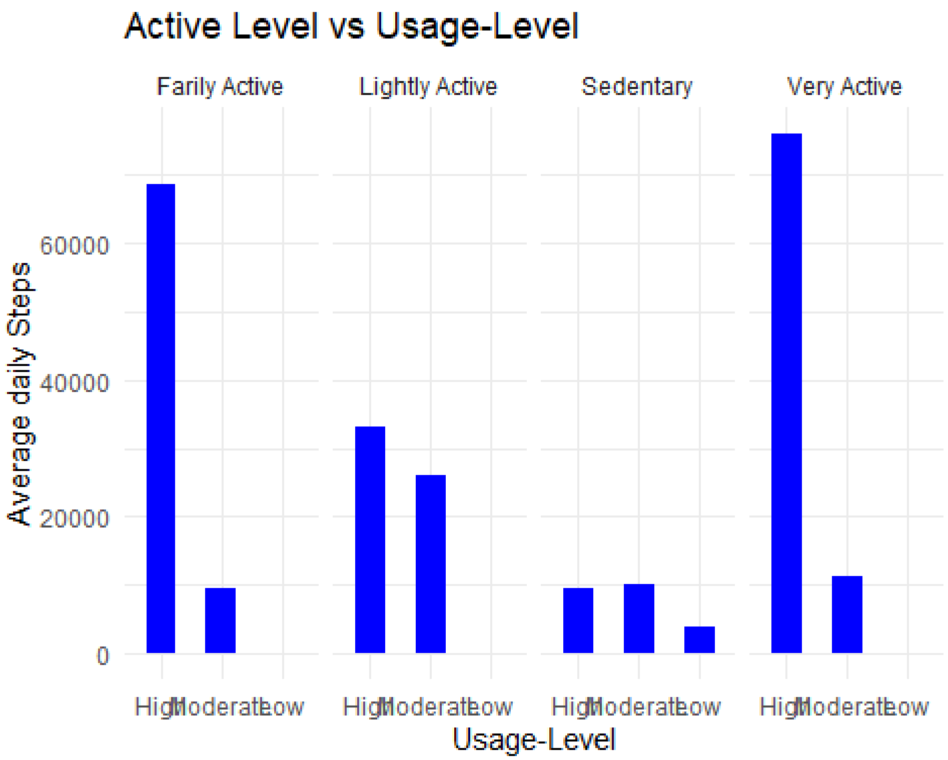


We can see that about 64% of the participating users were high usage of smart device and among these participants (21 users) there were about 28.6% very-active in taking the exercise with more than 10,000 steps in each day. About 38% of the high usage participants were fairly active in her daily activities with steps from 7500 to 10000 steps.

From the results of pie chart, the high-usage group of smart device with relatively low percentage of daily time at sedentary. We can infer from this result that longer time in wearing the smart device would lead to the user to become more active in moving the body and less time at sedentary.

Besides, the donut pie chart. I used the Facet_Grid() function to showcase the different usage level of the participant against different level of activity with in one visualization.

```
ggplot(usage_level_mean, aes(x= reorder(Usage_Level, -mean_daily_steps), y=
mean_daily_steps))+
  geom_bar(stat="identity", fill = "Blue", width =0.5)+
  facet_grid(.~steps_category)+
  theme_minimal()+
  labs(
    title = "Active Level vs Usage-Level",
    x = "Usage-Level",
    y = "Average daily Steps")
```



Conclusion & Recommendation

From the results of the analysis, I believe that smart device could provide incentive for user to become more active in her daily activities. The results also shown that the more time wearing the device, the less time for the sedentary. BellaBeat app could be a good solution for those people whom aren't exercise enough and want to have more calories burned. The app could periodically send alerts to users to encourage activity and remind them to achieve the daily goal of 10,000 steps. Furthermore, the app could provide vital health and diet alters to user to continuously monitoring the health and diet condition of the user.

In addition to the periodic notification about health and diet information, BellaBeat can promote users to practice a more regular 8 hours sleeping habit to maintain a better health. From the results, the 24 participants with sleep records shown the average sleep time was less than 8 hours a day. BellaBeat could help user by setting up a desired time to go to sleep and receive a notification minutes before to prepare to sleep. The company can offer additional resources to help users to fall asleep faster by providing breathing exercise, advises, relaxing music, sleep techniques when upgrading the BellaBeat app in the future App development.

The dataset provided, however, was not large enough (about 30 participants) to draw very reliable conclusions due to the bias and the lack of important details such as the demographic, sex, and age details of users.
Since Bellabeat's target customer is young and adult women, it would be beneficial to

collect usage data from this group of user and to investigate any particular trends which might help to better market the products for this group of user.