How Can a Wellness Technology Company Play It Smart?

S NIDHIN

2022-04-26

About me:

Hi, Thank you for showing interest in my project. My name is S NIDHIN, a rookie analyst. I recently completed my Data Analytics certification by Google and as a part of the course, created my first project portfolio on "How Can a Wellness Technology Company(Bellabeat) Play It Smart?" . In this read you will find my way of approaching a situation and how I use my knowledge and skill to solve a business problem. Feel free to comment on the contact us page of my website. So let's dive into it!

Bellabeat

Bellabeat is the go-to wellness brand for women with an ecosystem of products and services focused on women's health. Founded by two enthusiastic entrepreneurs, Urška Sršen and Sando Mur, with a mission to empower women to reconnect with themselves, unleash their inner strengths and be what they were meant to be.

They develop wearable's and accompanying products that monitor biometric and lifestyle data to help women better understand how their bodies work and make healthier choices by collecting data on activity, sleep, stress, and reproductive health, which allows Bellabeat to empower women with knowledge about their own health and habits. Since it was founded in 2013, Bellabeat has grown rapidly and quickly positioned itself as a tech-driven wellness company for women.

End Goal

In this study, we will be focusing on finding insights to guide the marketing team to come up with a better and efficient marketing strategy for the company by analyzing few datasets. Our main objective will be to gain insight into how consumers are using their smart devices to track their daily activities, sleep, calorie exerted and more, based on which, we can better understand our audiences and penetrate deeply into the market for a scalable business.

Questions for the Analysis (Ask phase)

To reach the end goal, First, we needed to understand the data and then, the issues which bellabeat was facing, and to do that, we need to keep few questions in mind going forward.

- What are some trends in smart device usage?
- How could these trends help influence Bellabeat marketing strategy ?

  > In order to make effective recommendations on to how to improve/ optimize or even come up with a new marketing strategy, Bellabeat needs to understand the customer's needs based on the data collected on the usage of their smart fitness devices.

- Who are the main stakeholders ?

The main stakeholders are the two co-founders, **Urška Sršen**, Bellabeat's (Chief Creative Officer) and **Sando Mur**(Key member of the executive team). And also, Bellabeat **marketing analytics team**.

Business Task

Now, after completing the ask phase, we are now able to define the business task:

- To identify new growth opportunities.
- Study the customer's behaviour pattern while using their smart devices.
- Provide recommendations to come-up/optimize the marketing strategies.

Preparing the Data (Prepare Phase)

During the prepare phase,we used a publicaly available data set on Kaggle which was graciously suggested by Sršen. The name of the dataset is FitBit Fitness Tracker Data and feel free to explore.

We also confirmed the integrity and credibility of the data and that all the data was organised according to our needs. Then we proceeded to sorting and filtering of the data which you will see later in this read.

About Kaggle Dataset

This Kaggle data set was generated by respondents from a distributed survey via Amazon Mechanical Turk between 03.12.2016-05.12.2016. 30 eligible Fitbit users consented to the submission of personal tracker data, including minute-level output for physical activity, heart rate, and sleep monitoring. Individual reports can be parsed by export session ID (column A) or timestamp (column B). Variation between output represents use of different types of Fitbit trackers and individual tracking behaviors/preferences.

The owner of this database is Möbius

Provenance

SOURCES: < https://zenodo.org/record/53894#.X9oeh3Uzaao>

COLLECTION METHODOLOGY: Pre-processed

Now that we have our database, Let's explore this in RStudio.

Installing and loading common packages and libraries.

In order to explore these datasets we need to install few packages into our RStudio which will not only help us to analyse the data but also to clean, visualize and to make a .rmd file to share our work with others.

```
install.packages("tidyverse")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("lubridate")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("dplyr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("ggplot2")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("tidyr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("here")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("skimr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("janitor")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("rmarkdown")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("readr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
install.packages("gganimate")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

This might take a few minutes to execute, perfect time to grab a cup of coffee!

Once completed, we need to load these packages for us to use as demonstrated below

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

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

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

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

```
library(dplyr)
library(ggplot2)
library(tidyr)
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(rmarkdown)
library(readr)
```

Upload your CSV files to R

In this we will upload the relevant datasets that we had downloaded from kaggle into rstudio. The uploaded files will be visible in the The Files / Plots / Packages / Help / Viewer window in the bottom right side of your screen.

Loading your CSV files

Now, we will load the .csv file in r. During this phase we will also VIEW, CLEAN, FORMAT, and ORGANIZE the data as we go along. We found an issue with the data formats in few of the dataset which needs to be addressed if not, it will create problems during our analysis phase. We found the format of the dates were in *CHR* which needed to be **rectified** to *Date Format*. We corrected the format as you can see in the code chunk.

We'll create another dataframe for the Activity data.

```
daily_activity <- read_csv("dailyActivity_merged.csv",
    col_types = cols(Id = col_character(),
        ActivityDate = col_date(format = "%m/%d/%Y")))
head(daily_activity)
```

```
## # A tibble: 6 x 15
##   Id      ActivityDate TotalSteps TotalDistance TrackerDistance LoggedActivitie~
##   <chr>   <date>            <dbl>         <dbl>           <dbl>             <dbl>
## 1 150396~ 2016-04-12        13162          8.5             8.5                 0
## 2 150396~ 2016-04-13        10735          6.97            6.97                0
## 3 150396~ 2016-04-14        10460          6.74            6.74                0
## 4 150396~ 2016-04-15         9762          6.28            6.28                0
## 5 150396~ 2016-04-16        12669          8.16            8.16                0
## 6 150396~ 2016-04-17         9705          6.48            6.48                0
## # ... with 9 more variables: VeryActiveDistance <dbl>,
## #   ModeratelyActiveDistance <dbl>, LightActiveDistance <dbl>,
## #   SedentaryActiveDistance <dbl>, VeryActiveMinutes <dbl>,
## #   FairlyActiveMinutes <dbl>, LightlyActiveMinutes <dbl>,
```

```
## #   SedentaryMinutes <dbl>, Calories <dbl>
```

Once the data is loaded and correctly formatted, we will look into the basic's. Examine the contents of each rows and columns. We used the **str function** to view the contents of the data set. Then used the **colname function** to identify the column name which will be useful later-on

```
str(daily_activity)
```

```
## spec_tbl_df [940 x 15] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id                       : chr [1:940] "1503960366" "1503960366" "1503960366" "1503960366" ...
##  $ ActivityDate             : Date[1:940], format: "2016-04-12" "2016-04-13" ...
##  $ 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_character(),
##   ..   ActivityDate = col_date(format = "%m/%d/%Y"),
##   ..   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>
```

```
colnames(daily_activity)
```

```
##  [1] "Id"                       "ActivityDate"
##  [3] "TotalSteps"               "TotalDistance"
##  [5] "TrackerDistance"          "LoggedActivitiesDistance"
##  [7] "VeryActiveDistance"       "ModeratelyActiveDistance"
##  [9] "LightActiveDistance"      "SedentaryActiveDistance"
## [11] "VeryActiveMinutes"        "FairlyActiveMinutes"
## [13] "LightlyActiveMinutes"     "SedentaryMinutes"
## [15] "Calories"
```

We will perform the same procedure's for importing other datasets as well

We'll create another dataframe for the calories data.

```
daily_calories <- read_csv("dailyCalories_merged.csv",
    col_types = cols(Id = col_character(),
        ActivityDay = col_date(format = "%m/%d/%Y")))
head(daily_calories)
```

```
## # A tibble: 6 x 3
##   Id         ActivityDay Calories
##   <chr>      <date>         <dbl>
## 1 1503960366 2016-04-12      1985
## 2 1503960366 2016-04-13      1797
## 3 1503960366 2016-04-14      1776
## 4 1503960366 2016-04-15      1745
## 5 1503960366 2016-04-16      1863
## 6 1503960366 2016-04-17      1728
```

```
str(daily_calories)
```

```
## spec_tbl_df [940 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id         : chr [1:940] "1503960366" "1503960366" "1503960366" "1503960366" ...
##  $ ActivityDay: Date[1:940], format: "2016-04-12" "2016-04-13" ...
##  $ Calories   : num [1:940] 1985 1797 1776 1745 1863 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_character(),
##   ..   ActivityDay = col_date(format = "%m/%d/%Y"),
##   ..   Calories = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
colnames(daily_calories)
```

```
## [1] "Id"          "ActivityDay" "Calories"
```

We'll create another dataframe for the intensities data.

```
daily_intensities <- read_csv("dailyIntensities_merged.csv",
    col_types = cols(Id = col_character(),
        ActivityDay = col_date(format = "%m/%d/%Y")))
head(daily_intensities)
```

```
## # A tibble: 6 x 10
##   Id         ActivityDay SedentaryMinutes LightlyActiveMinutes FairlyActiveMinu~
##   <chr>      <date>                 <dbl>                <dbl>             <dbl>
## 1 1503960366 2016-04-12               728                  328                13
## 2 1503960366 2016-04-13               776                  217                19
## 3 1503960366 2016-04-14              1218                  181                11
## 4 1503960366 2016-04-15               726                  209                34
## 5 1503960366 2016-04-16               773                  221                10
## 6 1503960366 2016-04-17               539                  164                20
## # ... with 5 more variables: VeryActiveMinutes <dbl>,
## #   SedentaryActiveDistance <dbl>, LightActiveDistance <dbl>,
## #   ModeratelyActiveDistance <dbl>, VeryActiveDistance <dbl>
```

```
str(daily_intensities)
```

```
## spec_tbl_df [940 x 10] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
```

```
##  $ Id                       : chr [1:940] "1503960366" "1503960366" "1503960366" "1503960366" ...
##  $ ActivityDay              : Date[1:940], format: "2016-04-12" "2016-04-13" ...
##  $ SedentaryMinutes         : num [1:940] 728 776 1218 726 773 ...
##  $ LightlyActiveMinutes     : num [1:940] 328 217 181 209 221 164 233 264 205 211 ...
##  $ FairlyActiveMinutes      : num [1:940] 13 19 11 34 10 20 16 31 12 8 ...
##  $ VeryActiveMinutes        : num [1:940] 25 21 30 29 36 38 42 50 28 19 ...
##  $ SedentaryActiveDistance : num [1:940] 0 0 0 0 0 0 0 0 0 0 ...
##  $ LightActiveDistance      : num [1:940] 6.06 4.71 3.91 2.83 5.04 ...
##  $ ModeratelyActiveDistance: num [1:940] 0.55 0.69 0.4 1.26 0.41 ...
##  $ VeryActiveDistance       : num [1:940] 1.88 1.57 2.44 2.14 2.71 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_character(),
##   ..   ActivityDay = col_date(format = "%m/%d/%Y"),
##   ..   SedentaryMinutes = col_double(),
##   ..   LightlyActiveMinutes = col_double(),
##   ..   FairlyActiveMinutes = col_double(),
##   ..   VeryActiveMinutes = col_double(),
##   ..   SedentaryActiveDistance = col_double(),
##   ..   LightActiveDistance = col_double(),
##   ..   ModeratelyActiveDistance = col_double(),
##   ..   VeryActiveDistance = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
colnames(daily_intensities)
```

```
##  [1] "Id"                       "ActivityDay"
##  [3] "SedentaryMinutes"         "LightlyActiveMinutes"
##  [5] "FairlyActiveMinutes"      "VeryActiveMinutes"
##  [7] "SedentaryActiveDistance"  "LightActiveDistance"
##  [9] "ModeratelyActiveDistance" "VeryActiveDistance"
```

We'll create another dataframe for the Heartrate data.

```
 heartrate <- read.csv("heartrate_seconds_merged.csv")
head(heartrate)
```

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

```
str(heartrate)
```

```
## 'data.frame':    2483658 obs. of  3 variables:
##  $ Id   : num  2.02e+09 2.02e+09 2.02e+09 2.02e+09 2.02e+09 ...
##  $ Time : chr  "4/12/2016 7:21:00 AM" "4/12/2016 7:21:05 AM" "4/12/2016 7:21:10 AM" "4/12/2016 7:21:1
##  $ Value: int  97 102 105 103 101 95 91 93 94 93 ...
```

```
colnames(heartrate)
```

```
## [1] "Id"    "Time"  "Value"
```

We'll create another dataframe for the sleep data.

```
sleep <- read.csv("sleepDay_merged.csv")
head(sleep)
```

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

```
str(sleep)
```

```
## 'data.frame':    413 obs. of  5 variables:
##  $ Id                : num  1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ SleepDay          : chr  "4/12/2016 12:00:00 AM" "4/13/2016 12:00:00 AM" "4/15/2016 12:00:00 AM" "
##  $ 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 ...
```

```
colnames(sleep)
```

```
## [1] "Id"                 "SleepDay"           "TotalSleepRecords"
## [4] "TotalMinutesAsleep" "TotalTimeInBed"
```

We'll create another dataframe for the weight data.

```
weight <- read.csv("weightLogInfo_merged.csv")
head(weight)
```

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

```
str(weight)
```

```
## 'data.frame':    67 obs. of  8 variables:
##  $ Id            : num  1.50e+09 1.50e+09 1.93e+09 2.87e+09 2.87e+09 ...
##  $ Date          : chr  "5/2/2016 11:59:59 PM" "5/3/2016 11:59:59 PM" "4/13/2016 1:08:52 AM" "4/21/2
```

```
##  $ WeightKg      : num  52.6 52.6 133.5 56.7 57.3 ...
##  $ WeightPounds  : num  116 116 294 125 126 ...
##  $ Fat           : int  22 NA NA NA NA 25 NA NA NA NA ...
##  $ BMI           : num  22.6 22.6 47.5 21.5 21.7 ...
##  $ IsManualReport: chr  "True" "True" "False" "True" ...
##  $ LogId         : num  1.46e+12 1.46e+12 1.46e+12 1.46e+12 1.46e+12 ...
```

```
colnames(weight)
```

```
## [1] "Id"             "Date"           "WeightKg"       "WeightPounds"
## [5] "Fat"            "BMI"            "IsManualReport" "LogId"
```

We'll create another dataframe for the steps per minutes data.

```
minute_step <- read.csv("minuteStepsNarrow_merged.csv")
head(minute_step)
```

```
##           Id        ActivityMinute Steps
## 1 1503960366 4/12/2016 12:00:00 AM     0
## 2 1503960366 4/12/2016 12:01:00 AM     0
## 3 1503960366 4/12/2016 12:02:00 AM     0
## 4 1503960366 4/12/2016 12:03:00 AM     0
## 5 1503960366 4/12/2016 12:04:00 AM     0
## 6 1503960366 4/12/2016 12:05:00 AM     0
```

We'll create another dataframe for the steps per hour data.

```
steps_per_hours<- read.csv("hourlySteps_merged.csv")
head(steps_per_hours)
```

```
##           Id          ActivityHour StepTotal
## 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
```

We'll create another dataframe for the Intensity per hour data.

```
intensity_per_hour<- read_csv("hourlyIntensities_merged.csv")
```

```
## Rows: 22099 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ActivityHour
## dbl (3): Id, TotalIntensity, AverageIntensity
##
## 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.
```

```
head(intensity_per_hour)
```

```
## # A tibble: 6 x 4
##           Id ActivityHour          TotalIntensity AverageIntensity
##        <dbl> <chr>                          <dbl>            <dbl>
## 1 1503960366 4/12/2016 12:00:00 AM             20            0.333
## 2 1503960366 4/12/2016 1:00:00 AM               8            0.133
## 3 1503960366 4/12/2016 2:00:00 AM               7            0.117
```

```
## 4 1503960366 4/12/2016 3:00:00 AM                0            0
## 5 1503960366 4/12/2016 4:00:00 AM                0            0
## 6 1503960366 4/12/2016 5:00:00 AM                0            0
```

Now we have imported the datasets we require to run our analysis. We can import more if required, later.

Cleaning the dataset (Process Phase)

During this phase we will be executing data cleaning practices by using *skimr* and *janitor* packages. Also some advance functions to change strings from *CHR* to *Date*.

Basics cleaning:

In this we will clean and organise the dataset for our analysis. The function we used were glimpse(), skim_without_charts and also clean the names of the data using clean_names().

During the cleaning process we made the following observations-

- We did not find any challenges with Activity, Calories and Intensities Dataset. But the formatting, I used clear formatting. For Data types, Dates columns needed to be converted to date type which we did while importing the dataset.

In R we have multiple ways to deal with data types and in order to demonstrate that, we intentionally left few dataset with incorrect date types which we will address to with advance data types conversion functions.

- In Sleep data : 3 duplicates were found and removed.
- In Weight data: We identified too many missing values in one column. So we decided to remove that column.

Advance Data type formating

Let's address the time stamp data before proceeding to analysis:

```
sleep$SleepDay=as.POSIXct(sleep$SleepDay, format="%m/%d/%Y %I:%M:%S %p", tz=Sys.timezone())
sleep$date <- format(sleep$SleepDay, format = "%m/%d/%y")
sleep$date=as.Date(sleep$date, "% m/% d/% y")
```

```
heartrate$Time=as.POSIXct(heartrate$Time, format="%m/%d/%Y %I:%M:%S %p", tz=Sys.timezone())
heartrate$date <- format(heartrate$Time, format = "%m/%d/%y")
heartrate$date=as.Date(heartrate$date, "% m/% d/% y")
```

```
weight$Date=as.POSIXct(weight$Date, format="%m/%d/%Y %I:%M:%S %p", tz=Sys.timezone())
weight$date <- format(weight$Date, format = "%m/%d/%y")
weight$date=as.Date(weight$date, "% m/% d/% y")
```

###Cleaning Minute Step dataset

```
#Cleaning Minute Step dataset
minute_steps_clean <- minute_step
minute_steps_clean$Id <- as.character(minute_steps_clean$Id)
minute_steps_clean$ActivityMinute <- mdy_hms(minute_steps_clean$ActivityMinute)

head(minute_steps_clean)
```

```
##           Id      ActivityMinute Steps
## 1 1503960366 2016-04-12 00:00:00     0
## 2 1503960366 2016-04-12 00:01:00     0
## 3 1503960366 2016-04-12 00:02:00     0
## 4 1503960366 2016-04-12 00:03:00     0
```

```
## 5 1503960366 2016-04-12 00:04:00     0
## 6 1503960366 2016-04-12 00:05:00     0
```

Converting Text format to Datetime format and then split data and time into seperate columns.

```
intensity_per_hour_clean<- intensity_per_hour
intensity_per_hour_clean$ActivityHour <- mdy_hms(intensity_per_hour_clean$ActivityHour)

#Splitting Date column in to two separate columns

intensity_per_hour_clean[c('Date', 'Time')] <- str_split_fixed(intensity_per_hour_clean$ActivityHour, "
intensity_per_hour_clean$Time<-  hms::as_hms(intensity_per_hour_clean$Time)
#view
head(intensity_per_hour_clean)
```

```
## # A tibble: 6 x 6
##           Id ActivityHour        TotalIntensity AverageIntensity Date      Time
##        <dbl> <dttm>                       <dbl>            <dbl> <chr>     <tim>
## 1 1503960366 2016-04-12 00:00:00             20            0.333 2016-04-~ 00:00
## 2 1503960366 2016-04-12 01:00:00              8            0.133 2016-04-~ 01:00
## 3 1503960366 2016-04-12 02:00:00              7            0.117 2016-04-~ 02:00
## 4 1503960366 2016-04-12 03:00:00              0            0     2016-04-~ 03:00
## 5 1503960366 2016-04-12 04:00:00              0            0     2016-04-~ 04:00
## 6 1503960366 2016-04-12 05:00:00              0            0     2016-04-~ 05:00
```

Summary (Analyze Phase)

Now that we have cleaned and organise our datasets, we can finally perform the data analysis and address our business tasks.

First, Let's check out how many unique participants are there in each dataframe? There might be variations in total participants in each dataframe

```
n_distinct(daily_activity$Id)
```

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

```
n_distinct(daily_calories$Id)
```

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

```
n_distinct(daily_intensities$Id)
```

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

```
n_distinct(heartrate$Id)
```

```
## [1] 14
```

```
n_distinct(sleep$Id)
```

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

```
n_distinct(weight$Id)
```

```
## [1] 8
```

As we can see, only daily_activity, daily_calories and daily_intensities are the only dataframe with consistent participants were as, heartrate has 14, sleep has 24 and weight has only 8 participants in them because of which we are not able to draw significant conclusion or make relevant recommendations.

We focused on Activity, Calories, Intensities and Sleep dataframe for now and summarize each dataframe to gain insights.

How many observations are there in each dataframe?

```
nrow(daily_activity)
```

```
## [1] 940
```

```
nrow(daily_calories)
```

```
## [1] 940
```

```
nrow(daily_intensities)
```

```
## [1] 940
```

```
nrow(sleep)
```

```
## [1] 413
```

We performed a quick summary statistics on each data frame.

For the daily activity dataframe:

```
daily_activity %>%
  select(TotalSteps,
         TotalDistance,
         SedentaryMinutes, Calories) %>%
  summary()
```

```
##    TotalSteps    TotalDistance    SedentaryMinutes    Calories
##  Min.   :    0   Min.   : 0.000   Min.   :   0.0   Min.   :   0
##  1st Qu.: 3790   1st Qu.: 2.620   1st Qu.: 729.8   1st Qu.:1828
##  Median : 7406   Median : 5.245   Median :1057.5   Median :2134
##  Mean   : 7638   Mean   : 5.490   Mean   : 991.2   Mean   :2304
##  3rd Qu.:10727   3rd Qu.: 7.713   3rd Qu.:1229.5   3rd Qu.:2793
##  Max.   :36019   Max.   :28.030   Max.   :1440.0   Max.   :4900
```

- We observed that, on an average, a participant take 7638 steps walking a 5.5Km(We are assuming the distance is in kilometes after doing a conversion) a day.

For the daily calories dataframe:

```
summary(daily_calories %>%
          select(Calories))
```

```
##     Calories
##  Min.   :   0
##  1st Qu.:1828
##  Median :2134
##  Mean   :2304
##  3rd Qu.:2793
##  Max.   :4900
```

For the Daily Intensities dataframe:

```
daily_intensities %>%
  select(SedentaryMinutes,LightlyActiveMinutes, FairlyActiveMinutes, VeryActiveMinutes) %>%
  summary()
```

```
##  SedentaryMinutes LightlyActiveMinutes FairlyActiveMinutes VeryActiveMinutes
```

```
##  Min.   :   0.0   Min.   :  0.0        Min.   :  0.00      Min.   :  0.00
##  1st Qu.: 729.8   1st Qu.:127.0        1st Qu.:  0.00      1st Qu.:  0.00
##  Median :1057.5   Median :199.0        Median :  6.00      Median :  4.00
##  Mean   : 991.2   Mean   :192.8        Mean   : 13.56      Mean   : 21.16
##  3rd Qu.:1229.5   3rd Qu.:264.0        3rd Qu.: 19.00      3rd Qu.: 32.00
##  Max.   :1440.0   Max.   :518.0        Max.   :143.00      Max.   :210.00
```

We have multiple observations here,

- The average Sedentary time is very high i.e. 16 hours.
- Majority of the participants are lightly active with an average of 3hrs a day.

For the sleep dataframe:

```
sleep %>%
  select(TotalSleepRecords,
  TotalMinutesAsleep,
  TotalTimeInBed) %>%
  summary()
```

```
##  TotalSleepRecords TotalMinutesAsleep TotalTimeInBed
##  Min.   :1.000     Min.   : 58.0      Min.   : 61.0
##  1st Qu.:1.000     1st Qu.:361.0      1st Qu.:403.0
##  Median :1.000     Median :433.0      Median :463.0
##  Mean   :1.119     Mean   :419.5      Mean   :458.6
##  3rd Qu.:1.000     3rd Qu.:490.0      3rd Qu.:526.0
##  Max.   :3.000     Max.   :796.0      Max.   :961.0
```

Here we see, an average participant spends about 30 mins in bed before falling asleep and then sleeps for 7 hours a day.

For the Weight dataframe:

```
weight %>%
  select(WeightKg, Fat, BMI) %>%
  summary()
```

```
##     WeightKg           Fat              BMI
##  Min.   : 52.60   Min.   :22.00   Min.   :21.45
##  1st Qu.: 61.40   1st Qu.:22.75   1st Qu.:23.96
##  Median : 62.50   Median :23.50   Median :24.39
##  Mean   : 72.04   Mean   :23.50   Mean   :25.19
##  3rd Qu.: 85.05   3rd Qu.:24.25   3rd Qu.:25.56
##  Max.   :133.50   Max.   :25.00   Max.   :47.54
##                   NA's   :65
```

Here we can see that the average weight of the participants is around 72Kgs with a BMI of 25.19 which puts majority of the participants in overweight category.

```
#Implicit missing values
minute_step %>%
  group_by(Id) %>%
  count() %>%
  filter(n < 44640) %>%
  arrange(desc(n))
```

```
## # A tibble: 33 x 2
## # Groups:   Id [33]
##            Id     n
```

```
##         <dbl> <int>
##  1 1624580081 44160
##  2 1927972279 44160
##  3 2026352035 44160
##  4 2873212765 44160
##  5 4558609924 44160
##  6 2022484408 44100
##  7 2320127002 44100
##  8 4388161847 44100
##  9 8053475328 44100
## 10 8877689391 44040
## # ... with 23 more rows
```

Incomplete observations. It should be 44,640 rows for each user Id (1440 * 31)

In Conclution

We have found the following insights from the given data sets,

- A participant takes 7638 steps per day by walking 5.5Km (We are assuming the distance is in kilometres after doing a conversion) , which is lower to what CDC recommends. CDC recommends that most adults should aim for 10,000 steps per day. For most people, this is the equivalent to 8 kilometres, or 5 miles.
- We found that a majority of participants are **lightly active**. The average Sedentary time is of 16 hours which is very high. This means that a majority of the participants are not into much physical activities rather engaging in sedentary behavior.
- We saw the sleeping pattern of the participants and observed that an average participant spends about **30 minutes in bed before falling asleep for 7 hours** a day.
- We can see that the average weight of the participants is around 72Kgs. Here we found a limitation with the dataframe, we cannot determine whether the participant were male or female because of which we cannot justify, whether the average weight is high or low. But with the **BMI of 25.19**, puts majority of the participants in **overweight category** putting them in high risk to diseases.

Merging these two datasets together

In order for us o find relationship between the activity level and sleeping pattern, we combained the two datasets together using the merge function.

```
combined_data1 <- merge(sleep, daily_activity, by="Id")
```

Take a look at how many participants are in this data set.

```
n_distinct(combined_data1$Id)
```

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

Note that there were more participant Ids in the daily activity dataset that have been filtered out using merge. So we considered using 'outer_join' to keep those in the dataset.

```
combained_data_outer1 <- merge(sleep, daily_activity, by= "Id", all = TRUE)
n_distinct(combained_data_outer1$Id)
```

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

Additional data required

```
combined_Intensity_Calories <- merge(daily_intensities,daily_calories, by="Id" , all =  TRUE)
n_distinct(combined_Intensity_Calories$Id)
```

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

```
combined_Intensity_Calories1 <- combined_Intensity_Calories %>%
  mutate(totalactivitytime= SedentaryMinutes+LightlyActiveMinutes+FairlyActiveMinutes+VeryActiveMinutes)
```

```
summary(combined_Intensity_Calories1)
```

```
##       Id             ActivityDay.x        SedentaryMinutes LightlyActiveMinutes
##  Length:27800       Min.   :2016-04-12   Min.   :   0.0   Min.   :  0.0
##  Class :character   1st Qu.:2016-04-19   1st Qu.: 728.0   1st Qu.:128.0
##  Mode  :character   Median :2016-04-26   Median :1055.0   Median :199.0
##                     Mean   :2016-04-26   Mean   : 988.4   Mean   :192.4
##                     3rd Qu.:2016-05-04   3rd Qu.:1225.0   3rd Qu.:262.0
##                     Max.   :2016-05-12   Max.   :1440.0   Max.   :518.0
##  FairlyActiveMinutes VeryActiveMinutes SedentaryActiveDistance
##  Min.   :  0.0       Min.   :  0.00    Min.   :0.000000
##  1st Qu.:  0.0       1st Qu.:  0.00    1st Qu.:0.000000
##  Median :  7.0       Median :  4.00    Median :0.000000
##  Mean   : 13.6       Mean   : 21.49    Mean   :0.001567
##  3rd Qu.: 19.0       3rd Qu.: 32.25    3rd Qu.:0.000000
##  Max.   :143.0       Max.   :210.00    Max.   :0.110000
##  LightActiveDistance ModeratelyActiveDistance VeryActiveDistance
##  Min.   : 0.000      Min.   :0.0000           Min.   : 0.000
##  1st Qu.: 1.960      1st Qu.:0.0000           1st Qu.: 0.000
##  Median : 3.370      Median :0.2400           Median : 0.210
##  Mean   : 3.349      Mean   :0.5681           Mean   : 1.522
##  3rd Qu.: 4.800      3rd Qu.:0.8000           3rd Qu.: 2.110
##  Max.   :10.710      Max.   :6.4800           Max.   :21.920
##  ActivityDay.y          Calories    totalactivitytime
##  Min.   :2016-04-12   Min.   :   0   Min.   :   2
##  1st Qu.:2016-04-19   1st Qu.:1827   1st Qu.: 989
##  Median :2016-04-26   Median :2156   Median :1440
##  Mean   :2016-04-26   Mean   :2313   Mean   :1216
##  3rd Qu.:2016-05-04   3rd Qu.:2800   3rd Qu.:1440
##  Max.   :2016-05-12   Max.   :4900   Max.   :1440
```

```
# Data check
 combined_Intensity_Calories1 %>%
  select(Id, totalactivitytime, Calories) %>%
  filter(totalactivitytime != 1440) %>%
  arrange(desc(totalactivitytime)) %>%
  group_by(Id) %>%
summary()
```

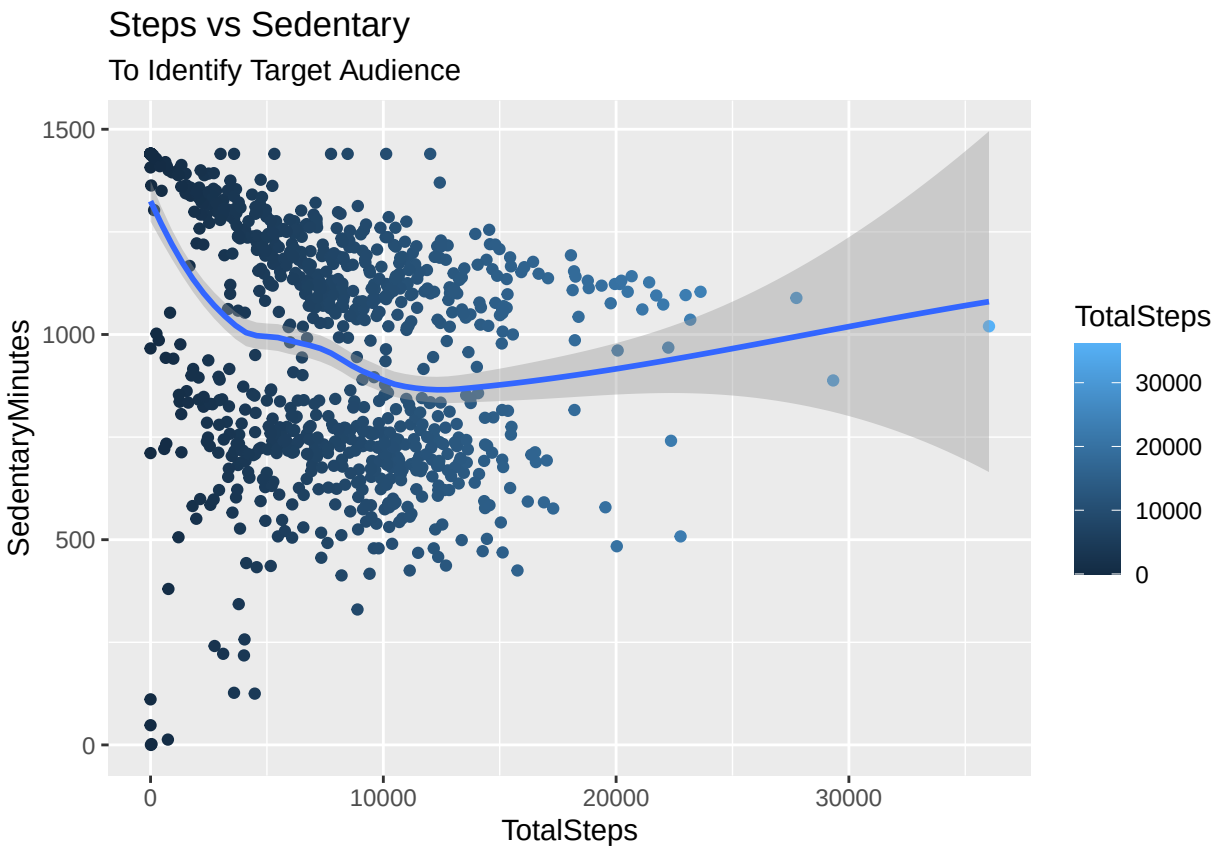
```
##       Id             totalactivitytime    Calories
##  Length:13844       Min.   :   2.0    Min.   :   0
##  Class :character   1st Qu.: 911.0    1st Qu.:1783
##  Mode  :character   Median : 989.0    Median :2158
##                     Mean   : 990.2    Mean   :2320
##                     3rd Qu.:1062.0    3rd Qu.:2859
##                     Max.   :1436.0    Max.   :4900
```

Plotting a few explorations

Here we will see a relationship between steps taken in a day and sedentary minutes to understand daily activity level.

```
ggplot(data=daily_activity, aes(x=TotalSteps, y=SedentaryMinutes, color= TotalSteps)) + geom_point()+ g
```

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



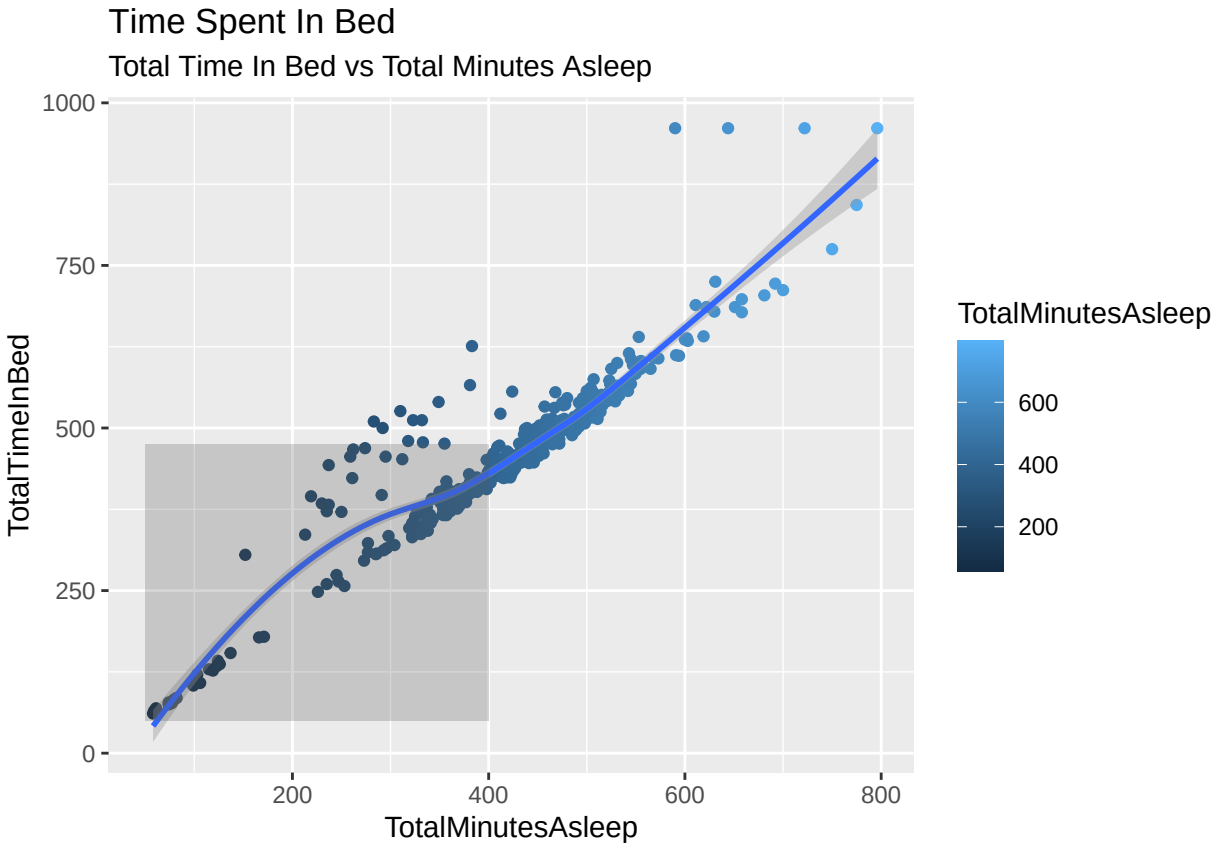
As you can see there is a negative co-relation between total number of steps taken and the total sedentary time. More the customer engages in sedentary behaviour, fewer steps are taken by them.

Here we have an opportunity to market ourselves by targeting these audiences which will encourage these audience to be more active throughout the day and thus increase their daily steps.

Here we will see the relationship between minutes asleep and time in bed to understand any trends.

```
ggplot(data=sleep, aes(x=TotalMinutesAsleep, y=TotalTimeInBed, color= TotalMinutesAsleep)) + geom_point
```

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



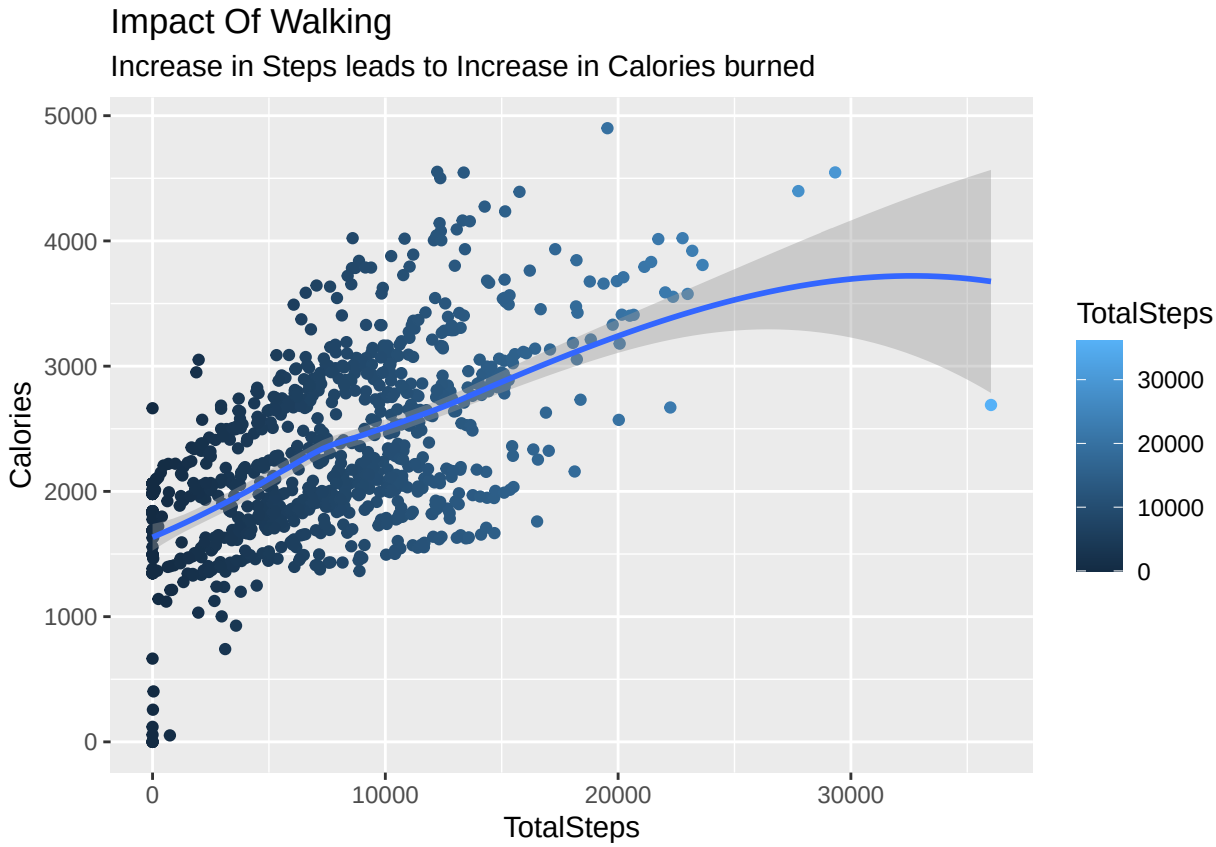
During our analysis, we found that a majority of the participants spend around 30 minutes in bed before falling asleep. In the above plot, we can conclude the same i.e. there is almost a linear trend between minutes asleep and time in bed as highlighted in darker shade.

This can occur due to various reasons such as reading, social media addiction, organizing one's thoughts, unhealthy lifestyle and more. We see an opportunity to improve our product by introducing push notifications or reminders to 'go to sleep' so the customers can improve their sleeping habits.

Relationship between steps and calories.

```
ggplot(data = daily_activity, aes(x= TotalSteps, y= Calories, col=TotalSteps ))+ geom_point()+ geom_smo
```

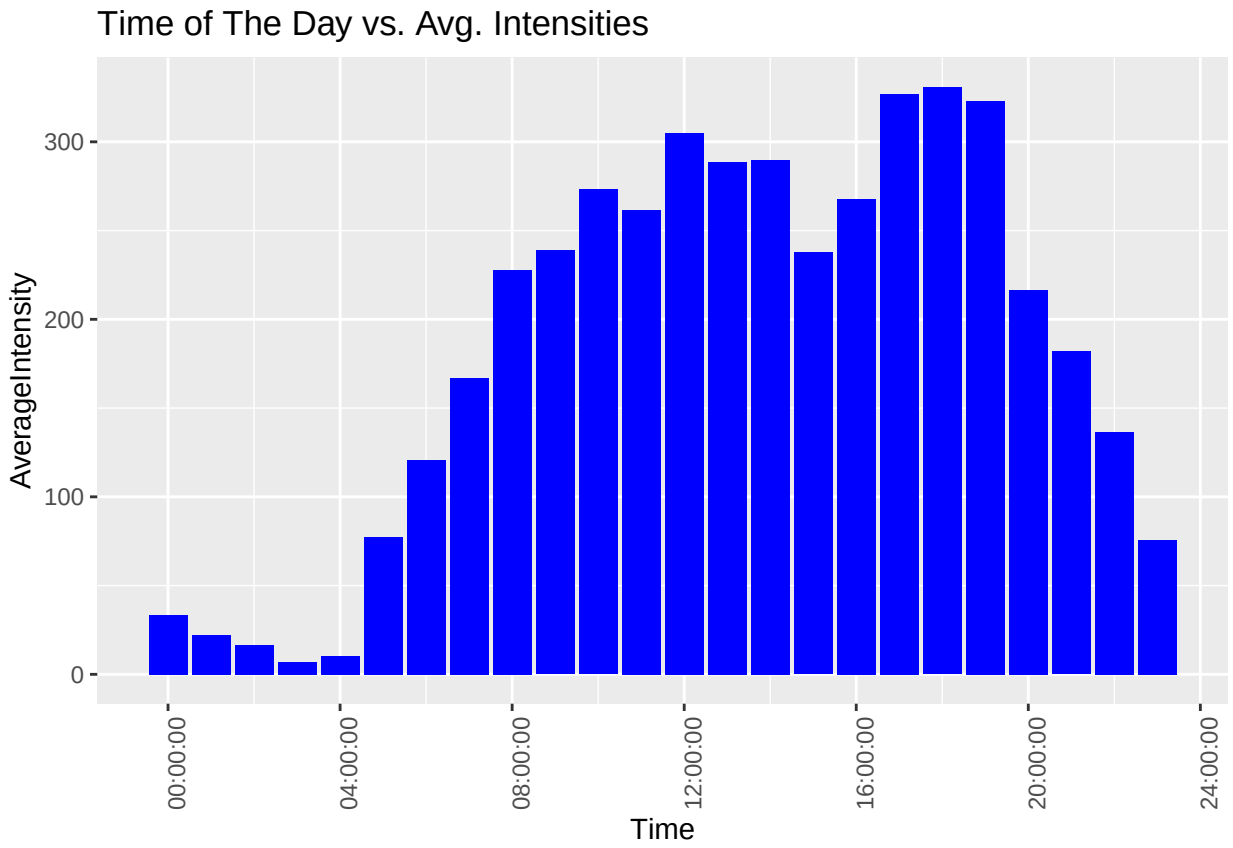
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



We can see a clear impact of increasing the activities as there is a positive co-relation between total steps taken and calories. The more active an individual will be more calories they'll likely to burn.

Intensities throughout the Day

```
ggplot(data=intensity_per_hour_clean, aes(x= Time , y= AverageIntensity))+geom_bar(stat = "identity", fi
  theme(axis.text.x = element_text(angle = 90))+
  labs(title = "Time of The Day vs. Avg. Intensities")
```

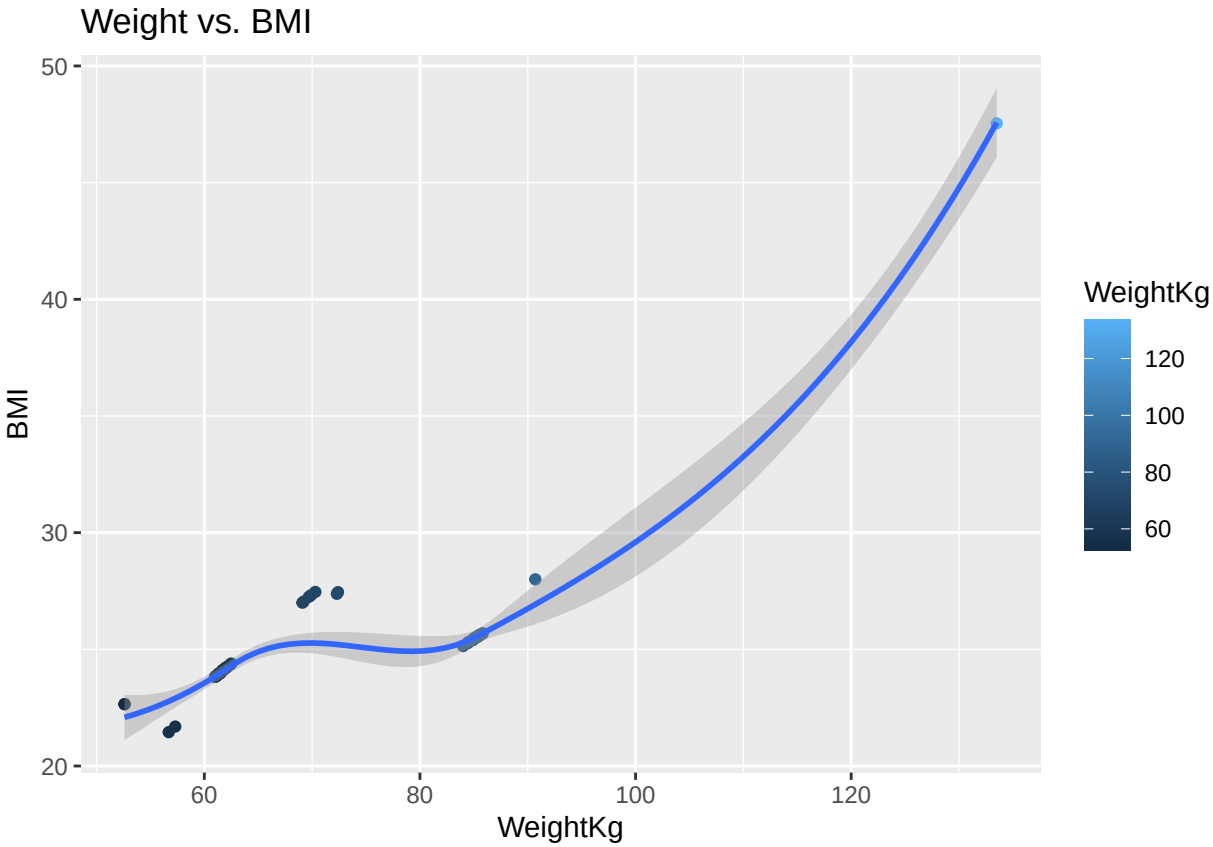


Here we can see the activity level of the participants throughout the day. They are mostly active between 5am to 11pm.During this time period they are moderately more active than the rest of the time period. They are most active in the evening between 5pm to 7pm and then decreased as we get closer to midnight.

Relationship Between Weights and BMI

```
ggplot(data = weight, aes(x=WeightKg, y= BMI, color=WeightKg ))+ geom_point()+geom_smooth()+ labs(title
```

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



As we can see, there is a positive co-relation between Weight of the participants when compared with their BMI percentage. As we found earlier, the average BMI of the participant is 25.19, which classifies them as overweight.

Findings

*The participants have a very low percentage to average steps per day compared to the CDC recommendations. More the participants engage in sedentary behavior, fewer steps are taken by them.

- The sedentary time of the participants is high and needs to be reduced.
- They spend around 30 minutes in bed before falling asleep for an average of 7hrs.
- Fewer calories are burned when steps per day decreases.
- They are highly active in the evening between 5pm and 7pm.
- A majority of the participants are classified as overweight's due to increased BMI

Recommendations

For Bellabeat Products.

- Introduce push notifications to motivate customers to shift to a better health.
- Notifications such as:

  -Stretching break in every few hours

  -Drinking water

  -Sleep Notification

-Exercise reminder

-Health Summary of the day containing steps taken, calories burned, BMI monitoring, etc.

- Work on UX/UI to increase customer engagement within bellabeat Eco-system.
- Recognizing daily activity pattern and providing reminder notifications to shift to a more active lifestyle.

Marketing

- Publish content about the benefits of a healthy active life.
- Blog Post- Providing insights on healthier choices in life. This will include eating habits, benefits of being active, importance of good sleep, and adverse effects of social media addiction and more.
- Segmented targeting, based on the usage of bellabeat products. Target people with high sedentary time. These audiences can be people with desk jobs, residential households, and more.
- The sleep data reviled that people spend around 30 minutes in bed before falling asleep, we can target these audiences to drive them toward benefits of a good night sleep or even provided sleep inducing music playlist.
- During the intensity analysis, we found people are fairly active during 5 to 11 am and 8 to 11 pm. We should choose this time period to run a campaign to bring in more customers. We should run this for a minimum of 3 days and with the data collected based on the traffic/lead conversion ratio, we can further optimise the campaign.
- We observed, when people took more steps a day, they burned more calories. We can target our audiences to improve their light activity habits. We can create milestones and push encouraging notification when these milestones are achieved. This will also help in reducing the BMI of an individual from overweight to healthy.
- Aggressive marketing on social media such as facebook, youtube, google, where people tends to spend most of their time.
- Re-targeting.
- Working on SEO with better content.

Additional Suggestions

- Bellabeat can come up with training programs for people to follow which should be published on their website and other social media handles.
- Introduction of diet plans for people to control their calorie intake and monitor calorie burned. This needs to be enjoyable thus making them sustainable in the long run.
- Create a summary of their weekly performance which will not only be shared with in the app itself but also as an email/ whatsapp notification.
- Email marketing of weekly newsletter and reminders.
- Referral Programs.

All our efforts should focus on how easy and convenient is bellabeat products for a customer to use and the benefits of a healthy lifestyle.

Feel free to reach out if you have any questions. Contact Us

Hope this was insightful to you.

Thanks.

S NIDHIN