Bellabeat Case Study

Sam Bowen

2023-07-10

Contents

Summary:

Bellabeat is an innovative high-tech company specializing in the manufacturing of health-focused smart products. Our range of smart devices gathers comprehensive data on various aspects including activity levels, sleep patterns, stress management, and reproductive health. By providing women with valuable insights into their own health and habits, Bellabeat aims to empower individuals to make informed decisions and take control of their well-being. Link to Bellabeat website: LINK

In this case study, our objective is to analyze usage data of non-Bellabeat smart devices to gain valuable insights into consumer behaviour. By examining how users interact with these devices, we aim to identify potential growth opportunities and enhance marketing strategies for Bellabeat. It is important to note that our analysis will specifically focus on the Bellbeat App as the central component of this study.

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 our line of smart wellness products.

Ask Phase:

Business Task:

The aim of this analysis is to examine usage date of non-Bellabeat smart devices to obtain valuable insights into how consumers engage with these devices. By understanding consumer behavior in relation to non-Bellabeat smart devices, we can enhance BellaBeat's marketing strategy and make informed decisions to further its success. ## Stakeholders: * Urška Sršen: Bellabeat's co-founder and Chief Creative Officer (CCO) * Sando Mur: Mathematician and Bellabeat's co-founder * Bellabeat Marketing Analytics Team:A team of data analysts responsible for collecting, analyzing, and reporting data that helps guide Bellabeat's marketing strategy.

Prepare Phase:

Data Used:

For this case study, we are utilizing FitBit Fitness Tracker Data as out data source, which is stored in Kaggle. The dataset, provided by Möbius, offers valuable insights into fitness tracking and forms the basis of our analysis.

Privacy & Accessibility of Data:

After reviewing the metadata for FitBit Fitness Tracker Data we confirmed that this is an open data source. This means Möbius has dedicated the work to the public domain by waiving all of his rights to the work worldwide under copyright law, including all related and neighboring rights, to the extent allowed by law. This allows you to copy, modify, distribute and perform the work, even for commercial purposes, all without asking permission. Link to resource

About Data:

The dataset used in this study was collected through a distributed survey conducted via Amazon Mechanical Turk between 03-12-2016 and 05-12-2016. A total of thirty eligible FitBit users coluntarily provided their personal tracker data, which includes minute-level information on physical activity, heart rate, and sleep monitoring, individual reports can be distinguished using the export session ID (column A) or timestamp (column B). The variations observed in the data reflect the usage of different FitBit trackers and individual tracking behaviors and preferences.Link to resource.

Organization & Verification of Data:

The dataset comprises 18 CSV files containing various quantitative data collected through Fitbit smart devices. Upon thorough review, it was determined that the data follows longitudinal structure, where each row represents a specific time stamp for an individual consumer. Consequently, each consumer is associated with multiple rows of data. Additionally, each consumer is identified by a unique identification number, and the data collected is organized based on day and time.

To ensure data accuracy, I conducted sorting, filtering, and created Pivot Tables in Excel. These actions allowed for meticulous analysis, identification of relationships, and confirmation of consistency across the different tables within the dataset.

Integrity & Credibilty of Data:

The dataset utilized in this case study lack demographic information about the users and has a relatively small sample size of only 30 users. Consequently, potential sampling bias may be present within the data. and we cannot ascertain whether the sample is representative of the entire population. To ensure an unbiased sample, it would be necessary to collect demographic information directly from FitBit.

Another limitation is that the dataset is outdated, being 7 years old, which implies that the information within it may not reflect current trends or behaviors accurately. Additionally, the data was collected over a limited period, specifically from 03-12-2016 to 05-12-2016, spanning only two months.

Considering these factors, we are adopting an operational approach in this case study, taking into account the constraints and potential limitations of the dataset.

Process Phase:

For this case study, I have opted to utilize the R programming language to conduct my analysis. R is an excellent choice for this task due to its exceptional capabilities in handling extensive datasets and generating sophisticated data visualizations. Its robust data processing capabilities will enable me to effectively explore and analyze the data, while its advanced visualization libraries will allow for clear and insightful presentations of the findings.

Installing Packages:

For this analysis I will be using the following packages in R:

- 'ggprepel'
- 'ggpubr'
- 'here'
- 'janitor'
- 'lubridate'
- 'skimr'
- 'tidyverse'

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```
install.packages("ggpubr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```
install.packages("ggrepel")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

Opening Libraries:

Next we will open the libraries.

```
library(ggpubr)
```

```
## Loading required package: ggplot2
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.2     v tibble    3.2.1
## v purrr     1.0.1     v tidyr     1.3.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
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)
library(ggrepel)
```

Importing Data:

Upon careful examination of the available data, we have selected three specific datasets for utilization and upload in this case study. The chosen datasets for analysis are as follows:

- 'Daily_activity'
- 'Daily_sleep'
- 'Hourly_steps'

```
daily_activity <- read_csv(file= "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(file= "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(file= "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.
```

Preview Uploaded Datasets:

To ensure the quality and suitability of our selected data frames, we will preview their contents and review the summary statistics for each column.

```
head(daily_activity)
```

```
## # A tibble: 6 x 15
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
##        <dbl> <chr>             <dbl>         <dbl>           <dbl>
## 1 1503960366 4/12/2016         13162          8.5             8.5
## 2 1503960366 4/13/2016         10735          6.97            6.97
## 3 1503960366 4/14/2016         10460          6.74            6.74
## 4 1503960366 4/15/2016          9762          6.28            6.28
## 5 1503960366 4/16/2016         12669          8.16            8.16
## 6 1503960366 4/17/2016          9705          6.48            6.48
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

```
str(daily_activity)
```

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

```
str(daily_sleep)
```

```
## spc_tbl_ [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:
##  $ 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)
```

```
## spc_tbl_ [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>
```

Cleaning and Formatting of Data:

With a more comprehensive understanding of our data structure, we will proceed to thoroughly process the data in order to identify and address any potential inconsistencies or errors that may be present.

Sample Size Verification:

Before delving into the cleaning process, our initial step is to examine the data for unique users. This evaluation will provide us with a clear understanding of the sample size and guide us as we proceed with the data analysis.

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

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

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

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

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

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

Upon observation, it is apparent that the sample size for the 'daily_sleep' dataset is smaller compared to both the 'hourly_steps' and 'daily_activity' datasets. Nevertheless, we will still incorporate the 'daily_sleep' data for practice and analysis in this case study.

Checking Data for Duplicates:

Having confirmed the sample size of the data, our next step is to examine whether any duplicates exist within the dataset.

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

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

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

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

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

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

Removing Duplicates & N/A's:

Now, we will proceed with the removal of duplicate and N/A values.

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

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

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

After completing the removal process, we conducted a thorough review to ensure that duplicates and N/A values were successfully eliminated from the datasets.

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

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

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

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

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

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

Cleaning & Renaming Columns:

In preparation for the upcoming data merging process, it is essential to ensure consistent formatting and syntax across the datasets. To achieve this, we will transform all column names in the datasets to lowercase. This adjustment will contribute to harmonizing the formatting and facilitating a smooth merging process.

```
clean_names(daily_activity)
```

```
## # A tibble: 940 x 15
##            id activity_date total_steps total_distance tracker_distance
##         <dbl> <chr>               <dbl>          <dbl>            <dbl>
##  1 1503960366 4/12/2016           13162           8.5              8.5
##  2 1503960366 4/13/2016           10735           6.97             6.97
##  3 1503960366 4/14/2016           10460           6.74             6.74
##  4 1503960366 4/15/2016            9762           6.28             6.28
##  5 1503960366 4/16/2016           12669           8.16             8.16
##  6 1503960366 4/17/2016            9705           6.48             6.48
##  7 1503960366 4/18/2016           13019           8.59             8.59
##  8 1503960366 4/19/2016           15506           9.88             9.88
##  9 1503960366 4/20/2016           10544           6.68             6.68
## 10 1503960366 4/21/2016            9819           6.34             6.34
## # i 930 more rows
## # i 10 more variables: logged_activities_distance <dbl>,
```

```
## #   very_active_distance <dbl>, moderately_active_distance <dbl>,
## #   light_active_distance <dbl>, sedentary_active_distance <dbl>,
## #   very_active_minutes <dbl>, fairly_active_minutes <dbl>,
## #   lightly_active_minutes <dbl>, sedentary_minutes <dbl>, calories <dbl>
```

```
daily_activity<- rename_with(daily_activity, tolower)

clean_names(daily_sleep)
```

```
## # A tibble: 410 x 5
##          id sleep_day total_sleep_records total_minutes_asleep total_time_in_bed
##       <dbl> <chr>                   <dbl>                <dbl>             <dbl>
##  1   1.50e9 4/12/201~                   1                  327               346
##  2   1.50e9 4/13/201~                   2                  384               407
##  3   1.50e9 4/15/201~                   1                  412               442
##  4   1.50e9 4/16/201~                   2                  340               367
##  5   1.50e9 4/17/201~                   1                  700               712
##  6   1.50e9 4/19/201~                   1                  304               320
##  7   1.50e9 4/20/201~                   1                  360               377
##  8   1.50e9 4/21/201~                   1                  325               364
##  9   1.50e9 4/23/201~                   1                  361               384
## 10   1.50e9 4/24/201~                   1                  430               449
## # i 400 more rows
```

```
daily_sleep<- rename_with(daily_sleep, tolower)

clean_names(hourly_steps)
```

```
## # A tibble: 22,099 x 3
##            id activity_hour         step_total
##         <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
##  7 1503960366 4/12/2016 6:00:00 AM           0
##  8 1503960366 4/12/2016 7:00:00 AM           0
##  9 1503960366 4/12/2016 8:00:00 AM         250
## 10 1503960366 4/12/2016 9:00:00 AM        1864
## # i 22,089 more rows
```

```
hourly_steps<- rename_with(hourly_steps, tolower)
```

Consistency of Columns:

Having successfully verified and converted all column names to lowercase, we can now proceed with data cleaning to facilitate the merging process. Our initial focus will be on standardizing the date-time format in both the 'daily-activity' and 'daily-sleep' datasets to ensure consistency. Specifically, for the 'daily_sleep' data, which includes unnecessary time information, we will utilize the as_date function to extract and retain on the date component.

```
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"))
```

Conducted a thorough review of the cleaned datasets to ensure that the cleaning process was executed successfully.

```
head(daily_activity)
```

```
## # A tibble: 6 x 15
##           id date       totalsteps totaldistance trackerdistance
##        <dbl> <date>          <dbl>         <dbl>           <dbl>
## 1 1503960366 2016-04-12      13162          8.5             8.5
## 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
## # i 10 more variables: loggedactivitiesdistance <dbl>,
## #   veryactivedistance <dbl>, moderatelyactivedistance <dbl>,
## #   lightactivedistance <dbl>, sedentaryactivedistance <dbl>,
## #   veryactiveminutes <dbl>, fairlyactiveminutes <dbl>,
## #   lightlyactiveminutes <dbl>, sedentaryminutes <dbl>, calories <dbl>
```

```
head(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
```

Now, we will shift our focus to the 'hourly__steps' dataset. To ensure compatibility and consistency, we will employ the date_time function to convert the corresponding data into the desired format.

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

Conducted a thorough review of the cleaned datasets to ensure that the cleaning process was executed successfully.

```
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
```

Merging of Data:

Having completed the data cleaning process, we will proceed to merge the 'daily_activity' and 'daily_sleep' datasets to explore potential correlations. The primary keys used for merging will be the ID and Date columns, ensuring a comprehensive and accurate consolidation of the data.

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

Checking merged data to confirm it executed successfully.

```
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, ~
```

Analyse & Share Phase:

In this case study, our objective is to analyze prevalent trends among FitBit users and assess how this knowledge can be leveraged to enhance Bellabeat's Marketing Strategy in the future.

Categorizing Users:

Given the absence of demographic variables within our sample, our goal is to ascertain the user type using the available data. By analyzing the daily step count, we can classify users based on their activity levels. The user categorization would be as follows:

- Sedentary - Less than 5,000 steps a day (0-4,999 steps)
- Lightly Active - Between 5,000 and 7,499 steps a day
- Fairly Active - Between 7,500 and 9,999 steps a day
- Very Active - More than 10,000 steps a day

These classifications are based around the Graduated Step Index. Link to Article

To begin, we will calculate the average number of daily steps per user.

```
daily_average <- daily_activity_sleep %>%
  group_by(id) %>%
  summarise (average_daily_steps = mean(totalsteps), average_daily_calories = mean(calories), average_da
```

Check calculated data.

```r
head(daily_average)
```

```
## # A tibble: 6 x 4
##           id average_daily_steps average_daily_calories average_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
```

Next, we will categorize our users based on their daily average step count.

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

Check 'user_type'.

```r
head(user_type)
```

```
## # A tibble: 6 x 5
##           id average_daily_steps average_daily_calories average_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
## # i 1 more variable: user_type <chr>
```

With the inclusion of a new column representing the user type, we will proceed to construct a data frame that displays the percentage distribution of each user type. This will facilitate a clearer visualization of the user types on a graph.

```r
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("Very Active", "Fairly A
```

Checking 'user_type_percent'.

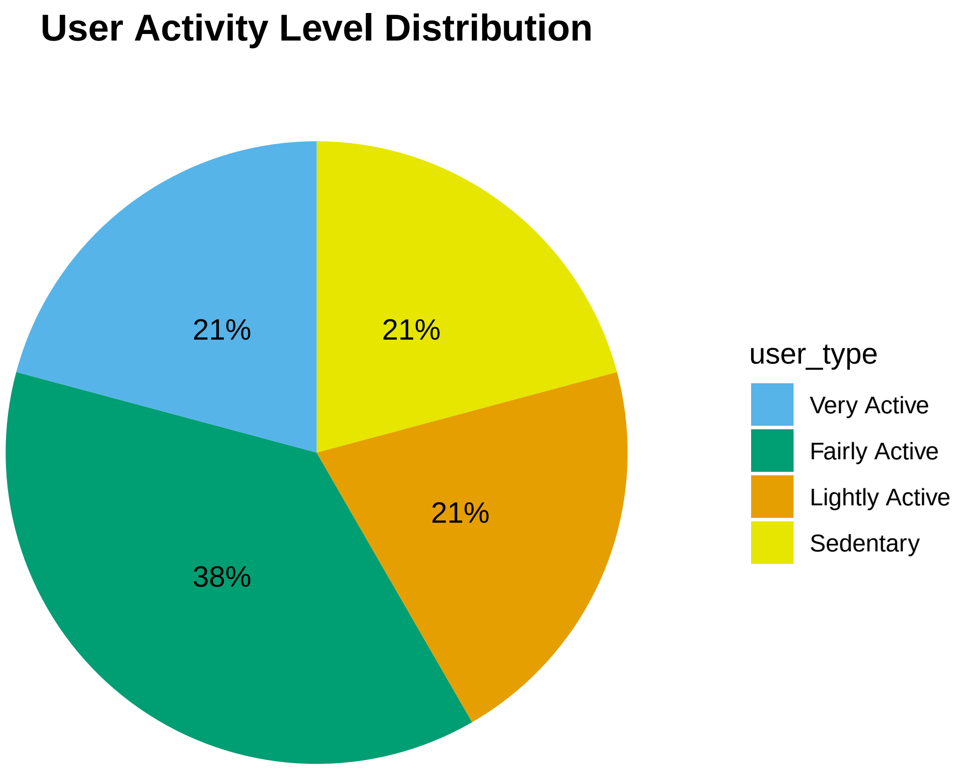
```r
head(user_type_percent)
```

```
## # A tibble: 4 x 3
##   user_type      total_percent labels
##   <fct>                  <dbl> <chr>
```

```
## 1 Fairly Active          0.375 38%
## 2 Lightly Active         0.208 21%
## 3 Sedentary              0.208 21%
## 4 Very Active            0.208 21%
```

The distribution of users based on their activity, as measured by the daily step count, indicates a fairly balanced representation across different users types.

```
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=14, face = "bold")) +
  scale_fill_manual(values = c("#56b4e9","#009e73", "#e69f00", "#e6e600")) +
  geom_text(aes(label = labels),
            position = position_stack(vjust = 0.5))+
  labs(title="User Activity Level Distribution")
```



This observation suggests that user of various kinds, irrespective of their activity levels, are utilizing smart devices.

Minutes Asleep & Steps per weekday:

Our objective now is to determine the day of the week when users are most active, as well as the days when users tend to sleep more. Additionally, we aim to verify whether users meet the recommended daily step count and achieve the recommended amount of sleep.

In the section below, we are performing calculations to derive the weekdays using the date column. Furthermore, we are determining the average number of steps walked and minutes slept for each weekday.

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

weekday_steps_sleep$weekday <-ordered(weekday_steps_sleep$weekday, levels=c("Monday", "Tuesday", "Wedne

weekday_steps_sleep <- weekday_steps_sleep %>%
  group_by(weekday) %>%
  summarize (daily_steps = mean(totalsteps), daily_sleep = mean(totalminutesasleep))
```

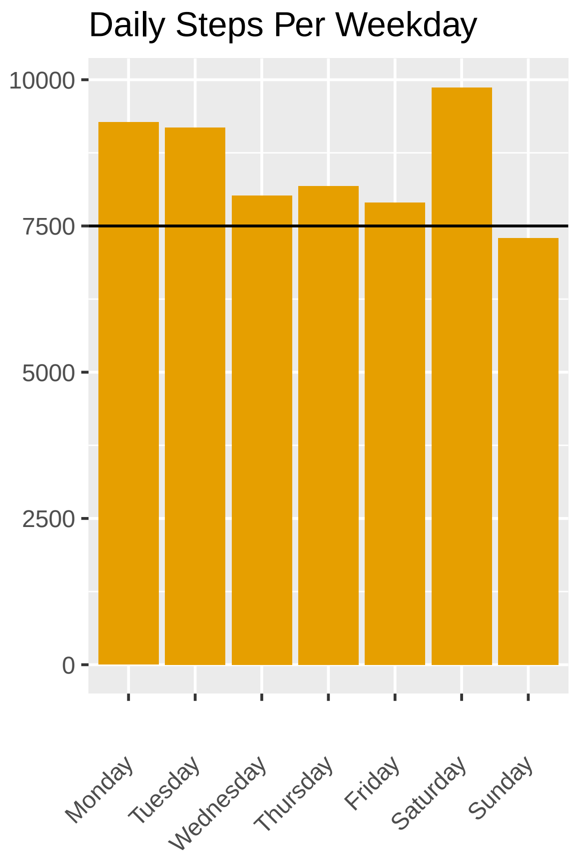
Checking 'weekday_steps_sleep'.

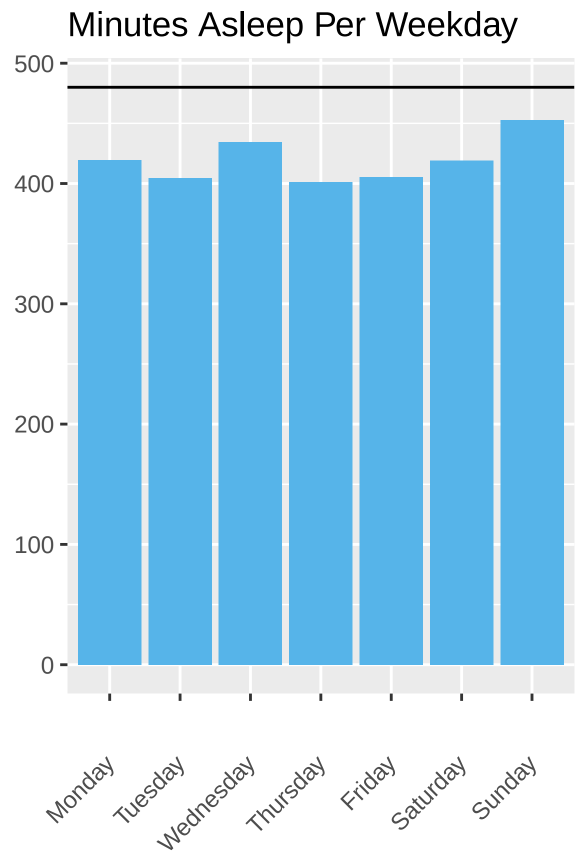
```
head(weekday_steps_sleep)
```

```
## # 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.
```

Creating visuals for daily steps per weekday and minutes asleep per weekday.

```
ggarrange(
  ggplot(weekday_steps_sleep) +
    geom_col(aes(weekday, daily_steps), fill = "#e69f00") +
    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)),
  ggplot(weekday_steps_sleep, aes(weekday, daily_sleep)) +
    geom_col(fill = "#56b4e9") +
    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)))
```





Based on the graphs presented above, we can see that users consistently achieve the recommended daily step count of 7,500, except on Sundays. In regards to daily sleep, we can see that users do not meet the recommended duration of sleep, which is typically 8 hours.

Hourly Steps

As we dive deeper into our analysis we aim to determine the specific time periods when users are most active throughout the day. For us to achieve this we will be utilizing the 'hourly_steps' data frame and information from the 'date_time' column.

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

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 934 rows [1, 25, 49, 73,
## 97, 121, 145, 169, 193, 217, 241, 265, 289, 313, 337, 361, 385, 409, 433, 457,
## ...].
```

Checking 'hourly_steps'.

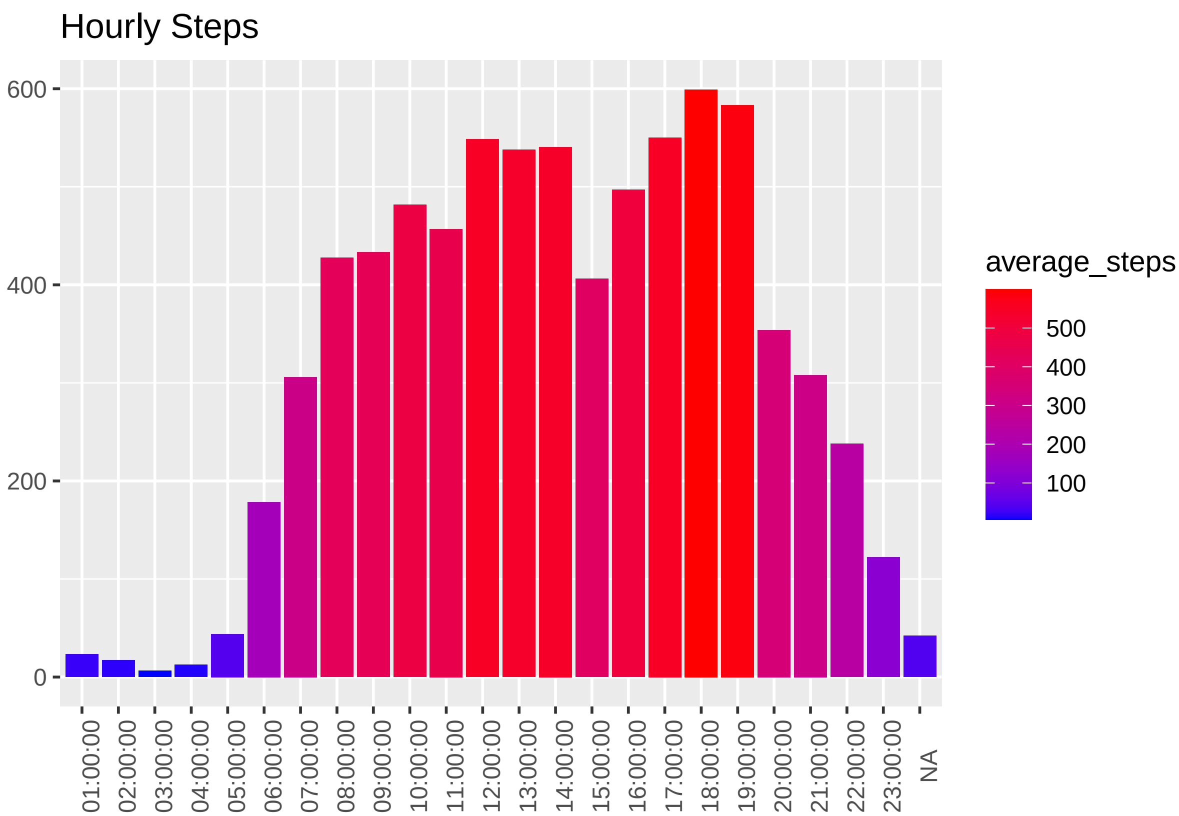
```
head(hourly_steps)
```

```
## # A tibble: 6 x 4
##           id date       time     steptotal
##        <dbl> <date>     <chr>        <dbl>
## 1 1503960366 2016-04-12 <NA>           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
```

Creating visualization for hourly steps throughout the day.

```
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", x="", y="") +
  scale_fill_gradient(low= "blue", high= "red") +
  theme(axis.text.x = element_text(angle = 90))
```



Our observation reveals that users exhibit higher levels of activity between 8am & 7pm. Particularly, a notable increase in step count is observed during lunchtime, specifically from 12pm to 2pm, as well as during the evening hours from 5pm to 7pm.

Correlations Between Data:

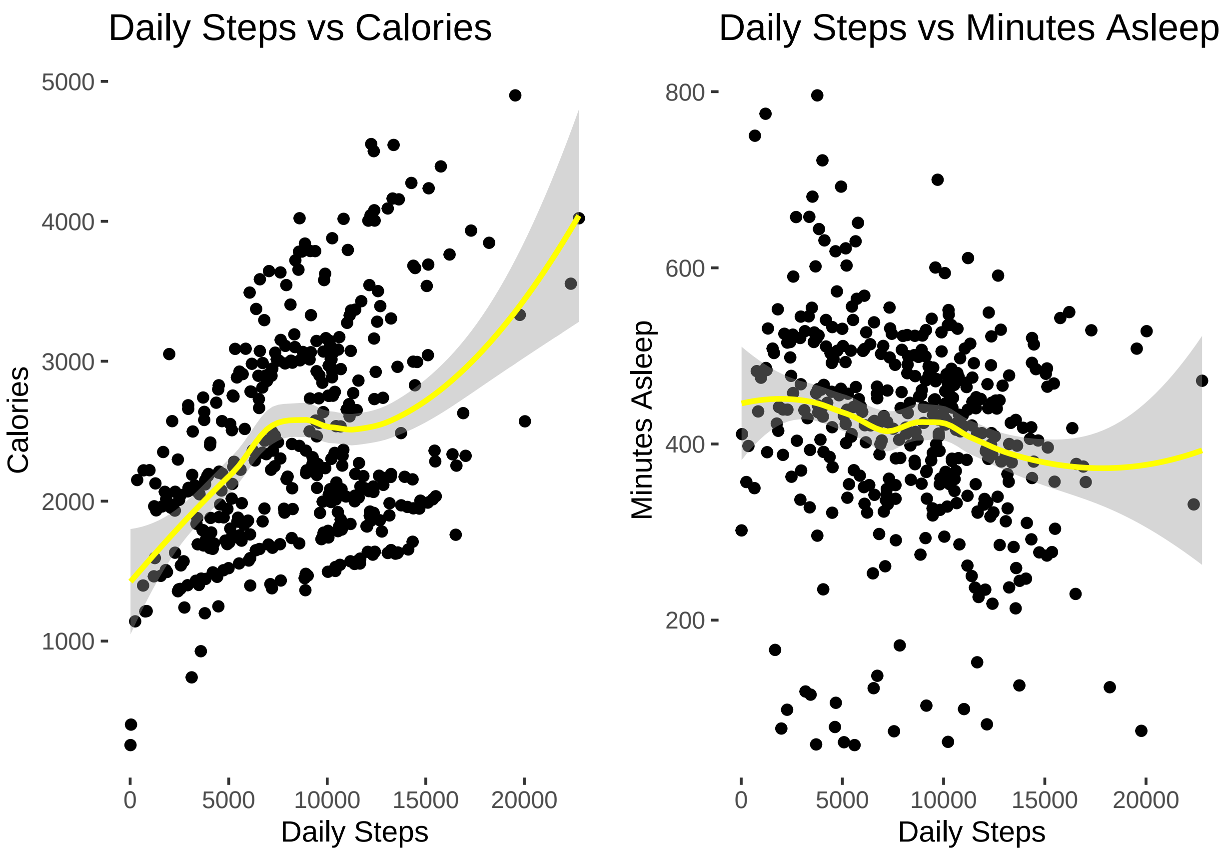
Next, we will assess whether any correlation exists between different variables, specifically:

- Daily Steps & Calories
- Daily Steps & Daily Sleep

```
ggarrange(
  ggplot(daily_activity_sleep, aes(x=totalsteps, y=calories)) +
    geom_jitter() +
    geom_smooth(color = "yellow") +
    labs(title = "Daily Steps vs Calories", x = "Daily Steps", y = "Calories") +
```

```
    theme(panel.background = element_blank(),
          plot.title = element_text(size=14)),
  ggplot(daily_activity_sleep, aes(x=totalsteps, y=totalminutesasleep)) +
    geom_jitter() +
    geom_smooth(color = "yellow") +
    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'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Based on the plots, the following observations can be made:

- There is no discernible correlation between the daily activity level, measured by steps and the duration of sleep in minutes. These variables appear to be independent of each other.
- However, a positive correlation is evident between the number of steps taken and the calories burned. As anticipated, a higher step count is associated with a greater number of calories burned.

Smart Device Use:

Smart Device Use: Days

Having observed certain patterns in activity, sleep, and calories burned, our next objective is to analyse the frequency of device usage among the users in our sample. The information will help us device an effective marketing strategy and identify which features would enhance the user experience with smart devices.

To ascertain the frequency of smart device usage within our sample over a 2 month period. we will calculate the number of users falling into three distinct categories:

- Low Use: User who make use of their device for a range of 1 to 10 days.
- Moderate use: Users who employ their device for a span of 10 to 20 days.
- High Use: Users who utilize their device for a duration of 21 to 31 days.

To start off, we will generate a new data frame by grouping the data based in the user ID. We will then calculate the number of days each user has used their smart device and incorporate a new column that corresponds to the aforementioned classification categories.

```
daily_use <- daily_activity_sleep %>%
  group_by(id) %>%
  summarize(days_used=sum(n())) %>%
  mutate(usage = case_when(
    days_used>= 1 & days_used <= 10 ~ "Low Use",
    days_used>= 11 & days_used <= 20 ~ "Moderate Use",
    days_used>= 21 & days_used <= 31 ~ "High Use",
  ))
```

Checking 'daily_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
```

To enhance the visualization of the results on a graph, we will proceed to construct a data frame that represents the percentages. Additionally, we will sort the usage levels in ascending order for better organization and clarity.

```
daily_use_percent <- daily_use %>%
  group_by(usage) %>%
  summarise(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
```

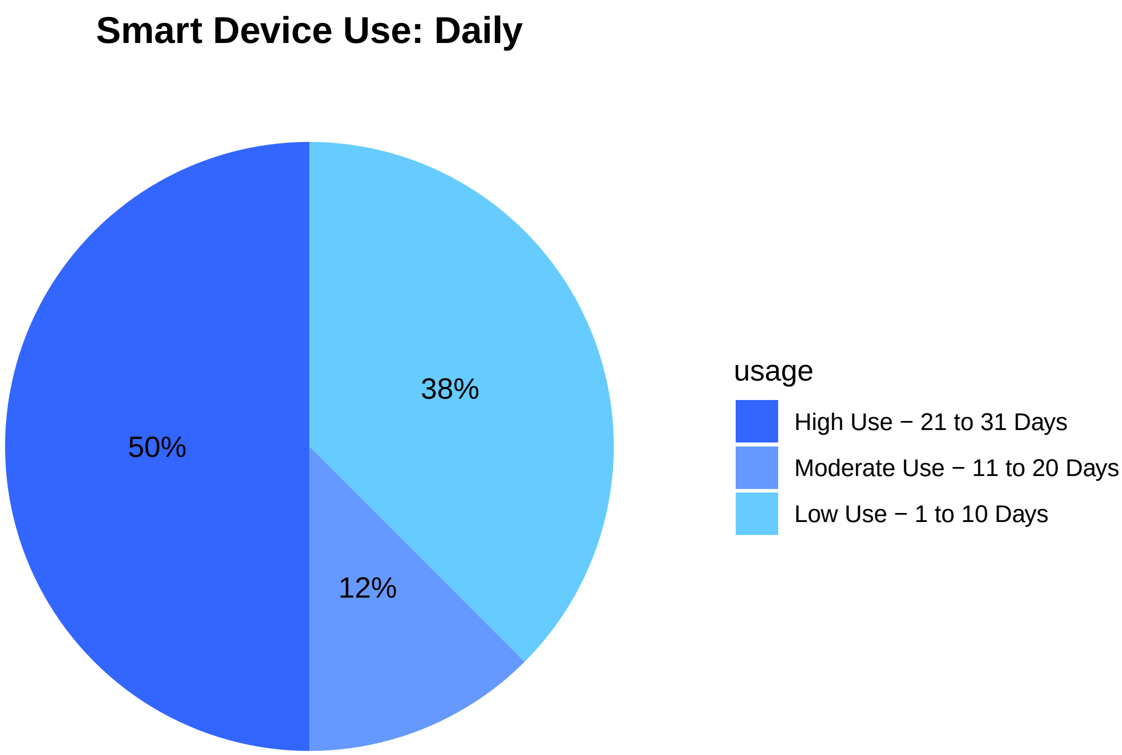
Checking 'daily_use_percent'.

```
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%
```

With our newly created table in hand, we can now proceed to create out plot.

```r
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_manual(values = c("#3366ff","#6699ff","#66ccff"),
                    labels = c("High Use - 21 to 31 Days", "Moderate Use - 11 to 20 Days", "Low Use - 1
  labs(title="Smart Device Use: Daily")
```



Upon Analyzing our results, it is evident that:

- Approximately 50% of the users in our sample exhibit frequent device usage, utilizing their devices for a duration of 21 to 31 days.
- Approximately 12 % of the users employ their devices for a span of 11 to 20 days.
- The remaining 38% of our sample demonstrate infrequent device usage.

Smart Device Use: Time

To gain more precise insights, we aim to examine the daily duration of device usage by users. To achieve this, we will merge the previously created 'daily_use' data frame with the 'daily_activity' data frame. This merger will enable us to filter the results based on the daily use of the device.

```r
daily_use_merged <- merge(daily_activity, daily_use, by=c("id"))
```

Checking 'daily_use_merged'.

```r
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
```

Next, we will generate a new data frame that calculates the total number of minutes users wore their device each day. Additionally, we will create three distinct categories based on the duration of device usage:

- Less than Half Day: Users who wore the device for less than half the day.
- More than Half Day: Users who wore the device for more than half the day.
- All Day: Users who wore the device for the entire day.

```r
minutes_worn <- daily_use_merged %>%
  mutate(total_minutes_worn = veryactiveminutes+fairlyactiveminutes+lightlyactiveminutes+sedentaryminut
  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"
```

```
  ))
```

Checking 'minutes_worn'.

```
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
```

As we have done previously, we will create new data frames to enhance the visualizations of our results. In this particular instance, we will generate four distinct data frames, which will be subsequently combined into a single visualization.

The first data frame will present the total number of users and calculate the percentage of device usage duration, considering the three previously defined categories.

The remaining three data frames will be filtered based on the categories of daily device users, allowing us to observe both the disparity in daily usage and the duration of device usage within each category.

```
minutes_worn_percent<- minutes_worn %>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
```

```r
  group_by(worn) %>%
  summarize(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) %>%
  summarize(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))

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

minutes_worn_lowuse<- minutes_worn %>%
  filter(usage == "Low Use") %>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(worn) %>%
  summarize(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))

minutes_worn_highuse$worn <- factor(minutes_worn_highuse$worn, levels = c("All Day","More Than Half Day
minutes_worn_percent$worn <- factor(minutes_worn_percent$worn, levels = c("All Day","More Than Half Day
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",
```

Checking 'minutes_worn_percent', 'minutes_worn_highuse', 'minutes_work_moduse', and 'minutes_work_lowuse'.

```r
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%
```

```r
head(minutes_worn_highuse)
```

```
## # A tibble: 3 x 3
##   worn               total_percent labels
##   <fct>                      <dbl> <chr>
## 1 All Day                   0.0676 6.8%
## 2 Less Than Half Day        0.0432 4.3%
```

```
## 3 More Than Half Day        0.889  88.9%
```

```
head(minutes_worn_moduse)
```

```
## # A tibble: 3 x 3
##   worn               total_percent labels
##   <fct>                      <dbl> <chr>
## 1 All Day                    0.267 27%
## 2 Less Than Half Day         0.04  4%
## 3 More Than Half Day         0.693 69%
```

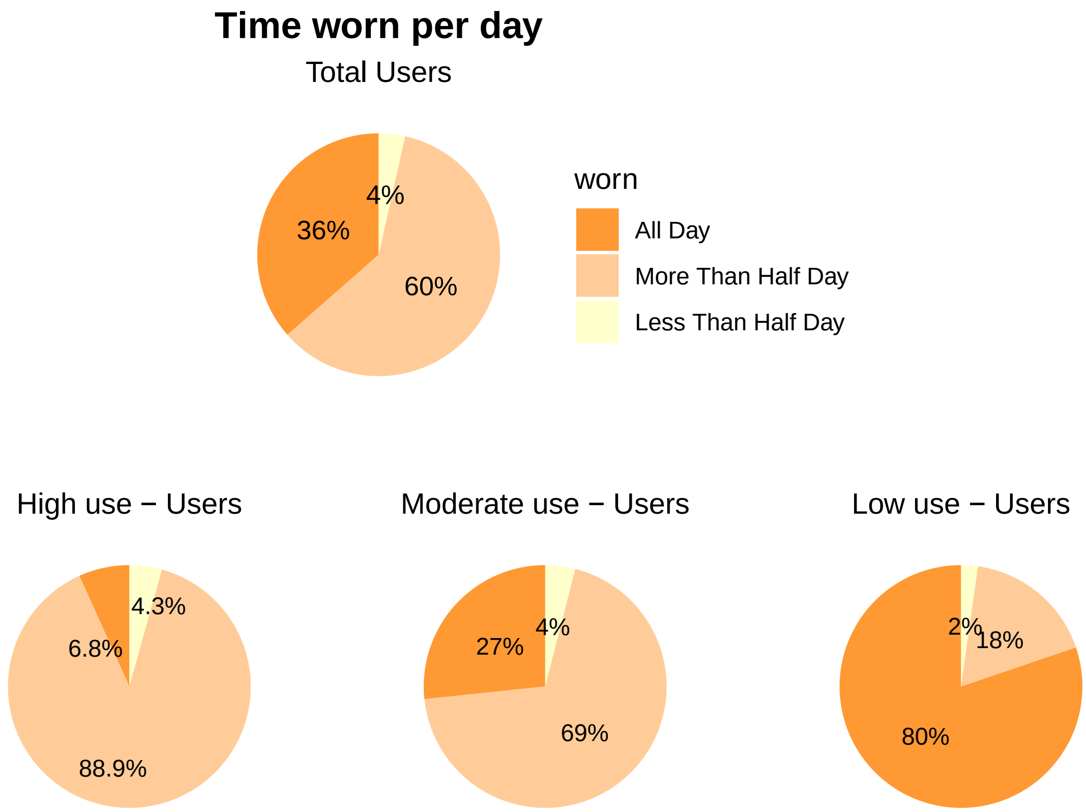
```
head(minutes_worn_lowuse)
```

```
## # A tibble: 3 x 3
##   worn               total_percent labels
##   <fct>                      <dbl> <chr>
## 1 All Day                   0.802  80%
## 2 Less Than Half Day        0.0224 2%
## 3 More Than Half Day        0.175  18%
```

With the creation of the four data frames and the arrangement of the worn level categories, we are now prepared to visualize our results through the following plots. To ensure optimal visualization, all plots have been consolidated together.

```
ggarrange(
  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_manual(values = c("#ff9933", "#ffcc99", "#ffffcc"))+
  geom_text(aes(label = labels),
            position = position_stack(vjust = 0.5), size = 3.5)+
  labs(title="Time worn per day", subtitle = "Total Users"),
  ggarrange(
  ggplot(minutes_worn_highuse, 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),
        legend.position = "none")+
    scale_fill_manual(values = c("#ff9933", "#ffcc99", "#ffffcc"))+
```

```
geom_text_repel(aes(label = labels),
          position = position_stack(vjust = 0.5), size = 3)+
labs(title="", subtitle = "High use - Users"),
ggplot(minutes_worn_moduse, 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),
      legend.position = "none") +
  scale_fill_manual(values = c("#ff9933", "#ffcc99", "#ffffcc"))+
geom_text(aes(label = labels),
          position = position_stack(vjust = 0.5), size = 3)+
labs(title="", subtitle = "Moderate use - Users"),
ggplot(minutes_worn_lowuse, 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),
      legend.position = "none") +
  scale_fill_manual(values = c("#ff9933", "#ffcc99", "#ffffcc"))+
geom_text(aes(label = labels),
          position = position_stack(vjust = 0.5), size = 3)+
labs(title="", subtitle = "Low use - Users"),
ncol = 3),
nrow = 2)
```



Based on our plotted data, it is evident that among the total number of users, 36% wear the device for the entire day, 60% wear it for more than half a day, only 4% wear it for less than half a day.

By further filtering the total users based on their device usage duration and examining the daily duration of device wear, we obtain the following results:

- In the case of High-Use Users who have utilized their device for a period of 21 to 31 days, only 6.85 wear the device throughout the entire day. However, 88.9% wear the device for more than half a day, though not the entire day.
- Moderate-Use Users tend to wear the device for shorter durations on a daily basis.
- Low-Use Users tend to wear their device for longer durations on the days they actually use it.

Act Phase (Conclusion)

At Bellabeat, our mission is to empower women to reconnect with themselves, unleash their inner strengths and be what they were meant to be.

To effectively address out business objective and assist Bellabeat in achieving their mission, I recommend leveraging their own tracking data for in-depth analysis. The datasets utilized in our current analysis have a limited sample size and potential biases due to the absence of target audience comprises young and adult women, I encourage further exploration of trends to devise a targeted marketing strategy tailored specifically to Bellabeat's needs.

Taking into consideration the insights gained from out analysis, we have identified various trends that hold potential to enhance our online marketing campaign and improve the Bellabeat App. Recommendations outlined below:

Engaging posts with updates & daily notifications in Bellabeat App:

- After classifying users into four categories, we observed that the majority of users walk over 7,500 steps per day, except on Sundays. To further motivate customers, we can encourage them to meet the CDC's daily recommended step count of 8,000 by sending alarms if they fall short and creating informative posts within our app that highlight the benefits of achieving this goal. According to the CDC, increased step count is associated with a lower mortality rate. Additionally, our analysis revealed a positive correlation between the number of steps taken and the calories burned.

Sleep Techniques & Sleep Notifications:

- Drawing insights from our findings, it is evident that users generally sleep less than the recommended 8 hours per day. To assist them in improving their sleep habits, we can offer a feature where users can set a desired bedtime and receive a notification a few minutes prior to that time, prompting them to prepare for sleep. Additionally, we can provide helpful resources within the app, such as breathing exercises, podcasts with relaxing music, and sleep techniques, to aid customers in achieving better quality sleep. By offering these resources, we aim to support users in establishing healthy sleep routines and enhancing their overall well-being.

Step's Club:

- Recognizing that notifications may not be motivating factor for everyone, we propose a steps club on the Bellabeat app. There would be 10,000 Steps Club, 12,500 Steps Club, and a 15,000 Steps Club for people that reach that amount of steps throughout the day. Users would get a digital badge that will show on their account so they can show it off and if they are able to have multiple days hitting 10,000 + steps they get points, which can be redeemed for merchandise or discounts on other Bellabeat products.

Challenges (Daily, Weekly, Monthly, Yearly, etc..):

- Everyone loves a challenge! Introducing a Daily, Weekly, Monthly, or even Yearly Steps challenge on the Bellabeat App. With the Steps Challenge, users have the freedom to create their own personalized challenge or join pre-existing challenges crafted within the app. Participants will strive to outdo each other and reach new heights of fitness achievement. As an added incentive, the winner of each challenge will be rewarded with exclusive merchandise and discounts on Bellabeat products.This dynamic feature empowers users to engage in friendly competition with friends, family, other app users, and even themselves, fostering a continuous drive to stay active and maintain a healthy lifestyle.

Rewards Program/Game:

- Can't forget about the gamers! Introduction of a game element withing the Bellabeat app for a limited duration. This game would involve progressing through different levels based on the daily step count. Users would need to sustain their activity level over a designated period, such as a week, to advance to the next level. Each level achievement would earn users a specific number of points, which can be redeemed for merchandise or discounts on other Bellabeat products. This gamer oriented approach aims to engage users and provide incentives for continued physical activity.

During our analysis, we not only examined trends in daily user habits but also discovered that only 50% of users utilize their device on a daily basis. Furthermore, we observed that merely 36% of users wear the device throughout the entire day when they use it. These findings provide valuable insights for promoting the features of Bellabeat's Products.

Armed with this information, we can continue to showcase the unique features and capabilities of Bellabeat's Products, emphasizing their potential to enhance user engagement and maximize the benefits of regular device usage. By highlighting the convenience, and value offered by Bellabeat, we can effectively reach out to user and encourage them to make the most of their device experience.

Citations:

- "Mobius" Kaggle, 2023, https://www.kaggle.com/arashnic/ Accessed 5 July 2023
- "FitBit Fitness Tracker Data" Kaggle, 2023, https://www.kaggle.com/datasets/arashnic/fitbit/ Accessed 5 July 2023
- Bellabeat, 2023, https://bellabeat.com/ Accessed 5 July 2023

- “CC0 1.0 Universal (CC0 1.0) Public Domain Dedication” Creative Commons, 2023, https://creativecommons.org/publicdomain/zero/1.0/ Accessed 5 July 2023
- Tudor-Locke C, Bassett DR Jr. How many steps/day are enough? Preliminary pedometer indices for public health. Sports Med. 2004;34(1):1-8. doi: 10.2165/00007256-200434010-00001. PMID: 14715035. https://pubmed.ncbi.nlm.nih.gov/14715035/