Bellabeat Case Study

Kevin Diep

1. Introduction

```
knitr::include_graphics("images/bellalogo.jpeg")
```

This project is a capstone case study for the Google Data Analytics Certificate.

Bellabeat is a high tech company that create health related product with the purpose of improving women health. Created in 2013, Bellabeat grew into a tech driven wellness company with a global consumer base. Their products and services emphasize the use of smart devices to monitor the user's activities, sleep, and stress. Urška Sršen, co founder and Chief Creative Officer of Bellabeat, believe analyzing data from a competitor's smart device, the FitBit Tracker will give new insights for company growth.

Here are the Bellabeat's product we would like to improve with description.

1. Bellabeat app: The Bellabeat app provides users with health data related to their activity, sleep, stress, menstrual cycle, and mindfulness habits. This data can help users better understand their current habits and make healthy decisions. The Bellabeat app connects to their line of smart wellness products.
2. Leaf: Bellabeat's classic wellness tracker can be worn as a bracelet, necklace, or clip. The Leaf tracker connects to the Bellabeat app to track activity, sleep, and stress.
3. Time: This wellness watch combines the timeless look of a classic timepiece with smart technology to track user activity, sleep, and stress. The Time watch connects to the Bellabeat app to provide you with insights into your daily wellness.
4. Spring: This is a water bottle that tracks daily water intake using smart technology to ensure that you are appropriately hydrated throughout the day. The Spring bottle connects to the Bellabeat app to track your hydration levels.
5. Bellabeat membership: Bellabeat also offers a subscription-based membership program for users. Membership gives users 24/7 access to fully personalized guidance on nutrition, activity, sleep, health and beauty, and mindfulness based on their lifestyle and goals.

For this case study, we will apply what we learned from the FitBit Tracker to improve the Bellabeats app.

Deliverables

1. How FitBit's data could give new insights on company's growth
2. Description of the FitBit Tracker Dataset along with it's reliability and limitation.
3. Documentation of the cleaning and/or manipulation of the datasets

4. Summary of our analysis
5. Collection of visualization that support the analysis, and key findings
6. Recommendation for next steps moving forward.

2. Ask

2.1 Business Task

Identify notable trends in the FitBit data and use our analysis to improve Bellabeat product.

2.2 Key Stakeholders

1. Co-founder & Chief Creative Officer: Urška Sršen
2. Co-founder & member of the Bellabeat executive team: Sando Mur
3. Bellabeat marketing analytics team

3. Prepare

3.1 DataSet

The dataset, FitBit Fitness Tracker Data, is provided by Mobius on the Kaggle platform. The data was obtained via survey by Amazon Mechanical Turk, a third party data source. The data consist of daily, hourly and minute information on user's daily activity, calories burned, sleep, and etc, all contained in 18 csv files. For this case study, we will focus on the daily and hourly data because they will provide more insight than the minute data. The parameter provided in the dataset line up with the ones provided by Information regarding dataset's columns can be found here (https://www.fitabase.com/media/1930/fitabasedatadictionary102320.pdf).

3.2 Limitation

The survey's sample size consist of only 33 users, so the data is not an accurate representation of the women customer base. This data was collected from time period 3/12/2016 to 5/12/2016 so the information is outdated.

3.3 Licensing and Privacy

This dataset is license under CC0: Public Domain thus freely available online for public use. This dataset may be copy, modify and distributed for any purpose without needing attribution or permission. All participants in the survey are anonymous.

4. Process

4.1 Libraries

```
library(dbplyr)
library(scales)
library(skimr)
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✓ ggplot2 3.4.0      ✓ purrr   0.3.5
## ✓ tibble  3.1.8      ✓ dplyr   1.0.10
## ✓ tidyr   1.2.1      ✓ stringr 1.4.1
## ✓ readr   2.1.3      ✓ forcats 0.5.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✕ readr::col_factor() masks scales::col_factor()
## ✕ purrr::discard()    masks scales::discard()
## ✕ dplyr::filter()     masks stats::filter()
## ✕ dplyr::ident()      masks dbplyr::ident()
## ✕ dplyr::lag()        masks stats::lag()
## ✕ dplyr::sql()        masks dbplyr::sql()
```

4.2 Import Dataset

```
daily_activity <- read.csv("Fitabase Data 4.12.16-5.12.16/dailyActivity_merged.csv")
daily_sleep <- read.csv("Fitabase Data 4.12.16-5.12.16/sleepDay_merged.csv")
heartrate_seconds <- read.csv("Fitabase Data 4.12.16-5.12.16/heartrate_seconds_merged.csv")
hourly_calories <- read.csv("Fitabase Data 4.12.16-5.12.16/hourlyCalories_merged.csv")
hourly_intensities <- read.csv("Fitabase Data 4.12.16-5.12.16/hourlyIntensities_merged.csv")
hourly_step <- read.csv("Fitabase Data 4.12.16-5.12.16/hourlySteps_merged.csv")
minutes_sleep <- read.csv("Fitabase Data 4.12.16-5.12.16/minuteSleep_merged.csv")
weight <- read.csv("Fitabase Data 4.12.16-5.12.16/weightLogInfo_merged.csv")

#Check unique ID
n_unique(heartrate_seconds$Id)
```

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

```
n_unique(weight$Id)
```

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

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

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

```
n_unique(minutes_sleep$Id)
```

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

Of the 18 csv files, we excluded 7 files containing only minute data and 3 files containing redundant information. We found the Weight and Heartrate data have a sample size of 8 and 14 people respectively. We can not draw any conclusion from a sample size of 14 and below. Thus we will exclude the Weight and Heartrate csv file. We kept the sleep csv file in because there are potentially useful insight within the data.

4.3 Merge Data

```
hourly_activity <- merge(
  merge(hourly_calories, hourly_intensities, by=c('Id'='Id', 'ActivityHour'='ActivityHour'), all
=TRUE),
  hourly_step, by=c('Id'='Id', 'ActivityHour'='ActivityHour'), all=TRUE)
```

4.4 Format Data

Task Performed

1. Change column names
2. Change data type of columns

```
daily_activity <- daily_activity %>%
  rename(Date = ActivityDate) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))

hourly_activity <- hourly_activity %>%
  rename(Datetime = ActivityHour) %>%
  mutate(Datetime = as.POSIXct(Datetime, format = "%m/%d/%Y %I:%M:%S %p"))
hourly_activity$Time <- format(hourly_activity$Datetime, format = "%H:%M:%S")
hourly_activity$Date <- as.Date(hourly_activity$Datetime, format = "%Y-%m-%d")


daily_sleep <- daily_sleep %>%
  rename(Date = SleepDay) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))

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

4.5 Check For Null Values or Duplicates

```
sum(duplicated(daily_activity[,1:2]))
```

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

```
sum(duplicated(hourly_activity[,1:2]))
```

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

```
sum(duplicated(daily_sleep[,1:2]))
```

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

```
sum(duplicated(minutes_sleep[,1:2]))
```

```
## [1] 543
```

```
is.null(daily_activity)
```

```
## [1] FALSE
```

```
is.null(hourly_activity)
```

```
## [1] FALSE
```

```
is.null(minutes_sleep)
```

```
## [1] FALSE
```

```
is.null(daily_sleep)
```

```
## [1] FALSE
```

```
daily_sleep <- daily_sleep %>%
  distinct(Id, Date, .keep_all = TRUE)

minutes_sleep <- minutes_sleep %>%
  distinct(Id, Datetime, .keep_all = TRUE)
```

We found daily sleep and minutes sleep data to have duplicate entries so we remove them. We check for null values within data and found none.

4.6 Additional Merging

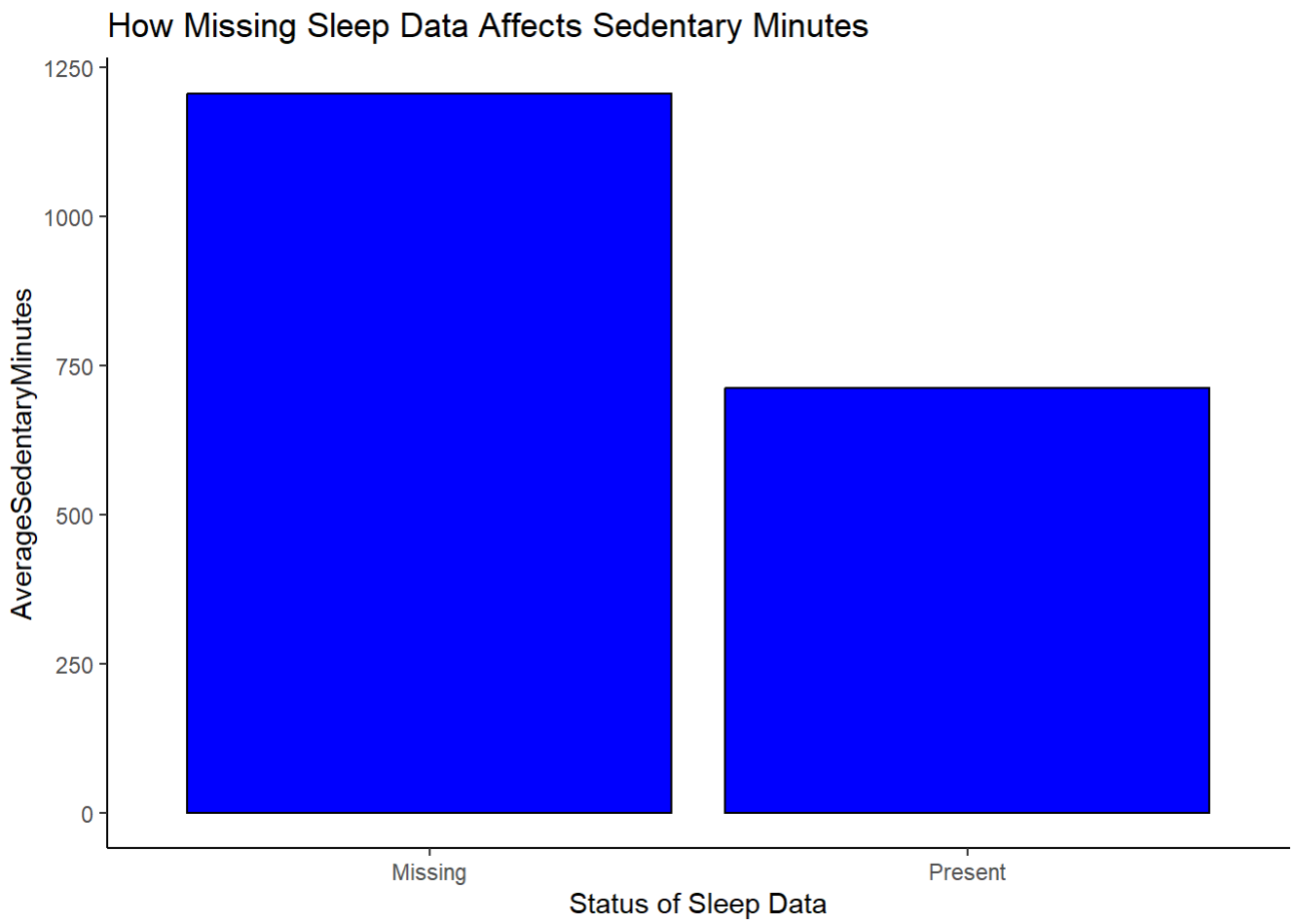
```
daily_activity <- merge(daily_activity, daily_sleep, by=c('Id'='Id', 'Date'='Date'), all.x=TRUE)
```

With columns now properly formatted and duplicated removed, we can merge daily activity table with daily sleep table. This new table contain some empty cells, but these empty cell simply indicate no sleep records were recorded that day.

5. Analyze & Share/Visualization

5.1 Comparing User Activity Levels

```
# Compare Sedentary Minutes column between rows with sleep data and rows without sleep data
modified_daily_activity <- daily_activity %>%
    select(Id, SedentaryMinutes, LightlyActiveMinutes, FairlyActiveMinutes, VeryActiveMinutes, T
otalMinutesAsleep, TotalTimeInBed, Calories) %>%
  mutate(TotalTimeBed = if_else(is.na(TotalTimeInBed), SedentaryMinutes + LightlyActiveMinutes +
FairlyActiveMinutes + VeryActiveMinutes, SedentaryMinutes + LightlyActiveMinutes + FairlyActiveM
inutes + VeryActiveMinutes + TotalTimeInBed)) %>%
  mutate(TotalTimeAsleep = if_else(is.na(TotalMinutesAsleep), SedentaryMinutes + LightlyActiveMi
nutes + FairlyActiveMinutes + VeryActiveMinutes, SedentaryMinutes + LightlyActiveMinutes + Fairl
yActiveMinutes + VeryActiveMinutes + TotalMinutesAsleep)) %>%
  mutate(HasSleepData = if_else(is.na(TotalMinutesAsleep), "0", "1")) %>%
  mutate(Equal1440 = if_else(TotalTimeAsleep == 1440, "1", "0"))

modified_daily_activity %>%
  group_by(HasSleepData) %>%
  summarise(AverageSedentaryMinutes = mean(SedentaryMinutes)) %>%
  ggplot(aes(x=HasSleepData, y=AverageSedentaryMinutes)) +
  geom_bar(stat="identity", fill="blue", color="black") +
  theme_classic() +
  labs(x="Status of Sleep Data", "Average Sedentary Minutes", title="How Missing Sleep Data Affe
cts Sedentary Minutes") +
  scale_x_discrete(labels = c("Missing","Present"))
```




```
# Percentage of data that adds up to 1440 minutes or 1 day
modified_daily_activity %>%
  filter(TotalTimeBed == 1440) %>%
  nrow() / 940 * 100
```

```
## [1] 64.25532
```

We discover an overlap in how Sedentary Minutes columns and Time Spent in Bed columns are recorded. Upon further inspection, we found entries without sleep data have larger Sedentary Minute values than the ones with sleep data. In addition, we summed up all the activity minutes including TotalTimeInBed column, to find majority of the entries add up to 1440 mins or 24 hours. Entries that add up to exactly 24 hours makes up 64.3% of the data. The other **35.7 % of the data contain total time both less than and greater than 1440 minutes**. This suggests a couple potential problems.

1. Entries less than 1440 minutes implies either the device is not recording properly to or the user's device was not active during at some point during the day. Distinction is difficult without further user input.
2. Entries larger than 1440 minutes should not be possible because this implies there are more than 24 hours being recorded in one day. Potential causes are: Mishandling of information, a system error that cause device to double count minute data or data from an adjacent day is being carried over.

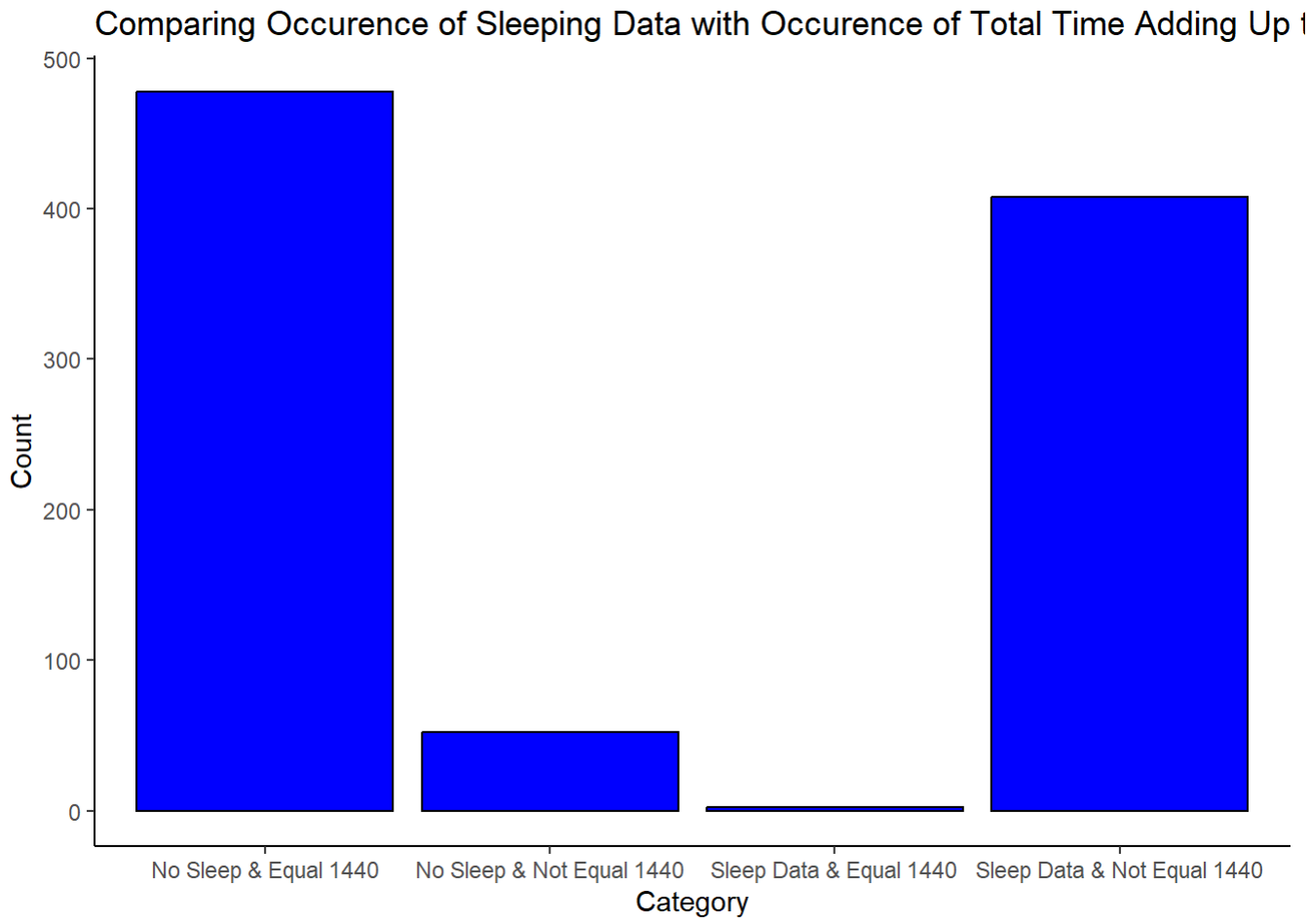
We can conclude **data without sleep data include their sleep time in Sedentary Minute while the ones with sleep data kept their sleep time separate from Sedentary Minute**. This would also explain why some Sedentary Minutes are abnormally high.

```
# TotalTimeBed column is the total for when the total time calculation use TotalTimeInBed
# TotalTimeAsleep column use TotalMinutesAsleep column instead of TotalTimeInBed
# Equal1440 checks if TotalTimeBed is equal to 1440
# Has SleepData check if sleep data is present

modified_daily_activity %>%
  select(TotalTimeBed, TotalTimeAsleep, HasSleepData, Equal1440)
```

TotalTimeBed <int>	TotalTimeAsleep <int>	HasSleepData <chr>	Equal1440 <chr>
1440	1421	1	0
1440	1417	1	0
1440	1440	0	1
1440	1410	1	0
1407	1380	1	0
1473	1461	1	0
1440	1440	0	1
1440	1424	1	0
1440	1423	1	0
1440	1401	1	0

1-10 of 940 rows Previous **1** 2 3 4 5 6 … 94 Next

```
# Check to see if there are any correlation between a the presence of sleep data and whether or
not it adds up to 1440 minutes
modified_daily_activity %>%
  select(Id, SedentaryMinutes, TotalTimeInBed, TotalMinutesAsleep, TotalTimeBed, TotalTimeAslee
p, HasSleepData, Equal1440) %>%
  group_by(HasSleepData, Equal1440) %>%
  tally() -> count_for_sleep

count_for_sleep$Name <- c("No Sleep & Not Equal 1440", "No Sleep & Equal 1440", "Sleep Data & No
t Equal 1440", "Sleep Data & Equal 1440")
count_for_sleep %>%
  rename(Count = n) %>%
  ggplot(aes(x=Name, y=Count)) +
  geom_bar(stat = "identity", color="black", fill ="blue") +
  theme_classic() +
  labs(x="Category", title="Comparing Occurence of Sleeping Data with Occurence of Total Time Ad
ding Up to 1440 Minutes")
```




```
# Calculate average time slept
daily_sleep %>%
  summarise(Average_Time_Slept = mean(TotalTimeInBed))
```

Average_Time_Slept
<dbl>
458.4829

1 row

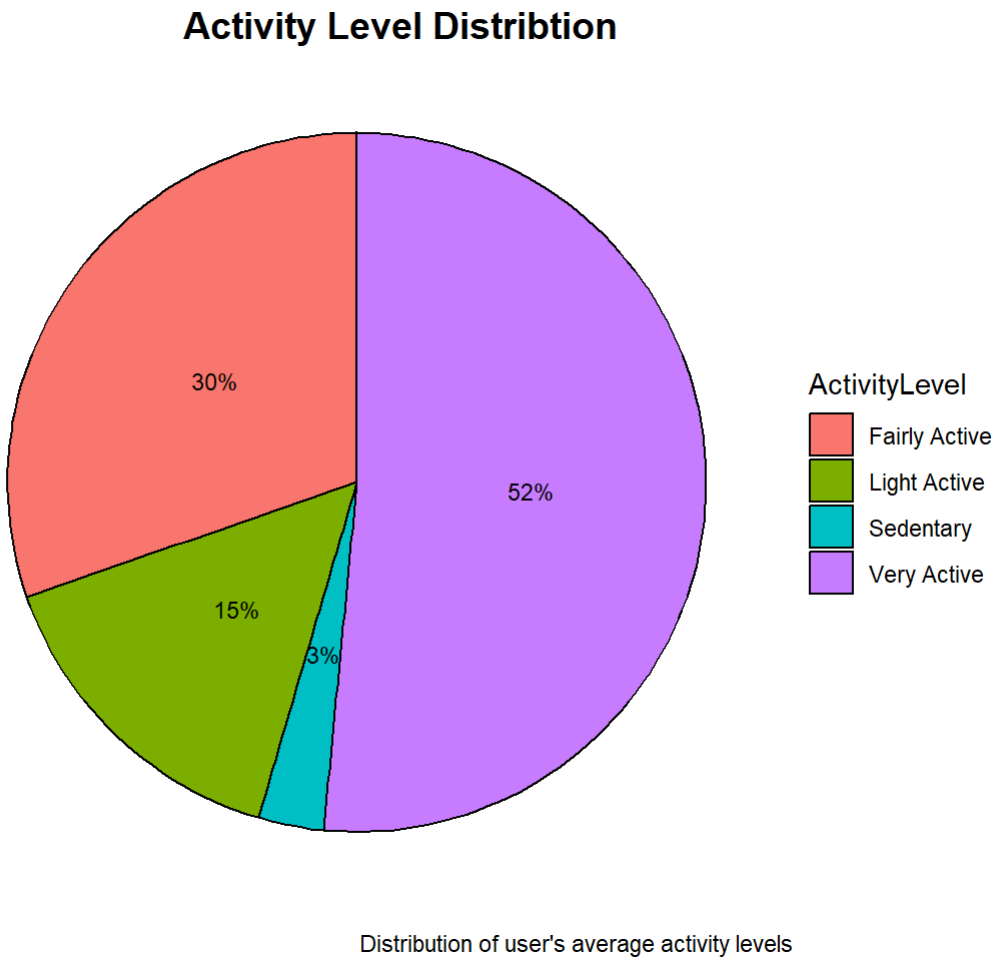
Next we check for a connection between presence of sleep data and total time equaling 1440 mins. First we need to check if TotalMinutesAsleep column would be a better fit than TotalMinuteinBed column in calculating total time. TotalTimeBed column use Total Time in Bed when summing up the day while TotalTimeAsleep column take TotalMinutesAsleep when summing up the day. We found no reoccurring value in TotalTimeAsleep except for days missing Sleep Data. Entries without sleep data will have equal TotalTimeBed and TotalTimeAsleep columns. Thus TotalMinutesInBed is the better measure here. Based on the Occurrence comparison bar graph above, we found that **often data without sleeping data will add up to 1440 minute while data with sleeping data won't add up to 1440 mins. This indicate the error lies in tracking sleep data. As a bonus, we found user on average spend 458.5 mins or roughly 7.7 hours asleep.**

```
# Categorize users in Very Active, Fairly Active, Lightly Active or Sedimentary Activity level
average_time_in_bed <- mean(daily_sleep$TotalTimeInBed)

activity_score_function <- function(sedentary, lightly, fairly, very) {
  average_activity_score <- (lightly + 2 * fairly + 3 * very)
  return(average_activity_score)
}

modified_daily_activity <- modified_daily_activity %>%
  mutate(ModifiedSedentaryMinutes = if_else(is.na(TotalTimeInBed), SedentaryMinutes - as.integer
(average_time_in_bed), SedentaryMinutes)) %>%
  select(Id, ModifiedSedentaryMinutes, LightlyActiveMinutes, FairlyActiveMinutes, VeryActiveMinu
tes) %>%
  group_by(Id) %>%
  summarise(Sedentary = mean(ModifiedSedentaryMinutes), Lightly = mean(LightlyActiveMinutes), Fa
irly = mean(FairlyActiveMinutes), Very = mean(VeryActiveMinutes))

activity_threshold <- modified_daily_activity %>%
  colMeans(modified_daily_activity$Lightly)

modified_daily_activity <- modified_daily_activity %>%
  mutate(Overall_Fitness = case_when(
    between(activity_score_function(Sedentary, Lightly, Fairly, Very), 0, 60) ~ "Sedentary",
    between(activity_score_function(Sedentary, Lightly, Fairly, Very), 61, 180) ~ "Light Activ
e",
    between(activity_score_function(Sedentary, Lightly, Fairly, Very), 181, 300) ~ "Fairly Activ
e",
    activity_score_function(Sedentary, Lightly, Fairly, Very) > 300 ~ "Very Active"
  ))

modified_daily_activity
```

Id	Sedentary	Lightly	Fairly	Very	Overall_Fitness
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>
1503960366	759.5161	219.93548	19.1612903	38.70967742	Very Active
1624580081	799.7419	153.48387	5.8064516	8.67741935	Fairly Active
1644430081	764.9333	178.46667	21.3666667	9.56666667	Fairly Active
1844505072	792.9355	115.45161	1.2903226	0.12903226	Light Active
1927972279	933.2903	38.58065	0.7741935	1.32258065	Sedentary
2022484408	654.5806	257.45161	19.3548387	36.29032258	Very Active
2026352035	645.0968	256.64516	0.2580645	0.09677419	Fairly Active
2320127002	776.8710	198.19355	2.5806452	1.35483871	Fairly Active
2347167796	610.8333	252.50000	20.5555556	13.50000000	Very Active
2873212765	639.1935	308.00000	6.1290323	14.09677419	Very Active

1-10 of 33 rows Previous **1** 2 3 4 Next

```
# Create pie plot based on user activity level
activity_minute_distribution <- data.frame(
  total = table(modified_daily_activity$Overall_Fitness),
  percentage = scales::percent(round(c(table(modified_daily_activity$Overall_Fitness)) / sum(tab
le(modified_daily_activity$Overall_Fitness)), 2))) %>%
  rename(ActivityLevel = total.Var1) %>%
  rename(Frequency = total.Freq)

ggplot(activity_minute_distribution, aes(x="", y=Frequency, fill=ActivityLevel)) +
  geom_bar(stat="identity", width=2, color="black") +
  coord_polar("y", start=0) +
  theme_void() +
  theme(plot.title = element_text(hjust = 0.6, size=14, face = "bold")) +
  geom_text(aes(label=percentage), color="black", size=3,  position = position_stack(vjust = 0.
5)) +
  labs(title="Activity Level Distribtion", caption="Distribution of user's average activity leve
ls")
```



Distribution of user's average activity levels

For this project, we consider daily activities that require some exertion such as walking or lifting as light activities. In addition we treated 1 minute of fairly active activity and 1 minute of very active activity as 2 and 3 minutes of light activity respectively. We classify people into the following:

1. Sedentary - Less than 1 hour of light activity per day on average

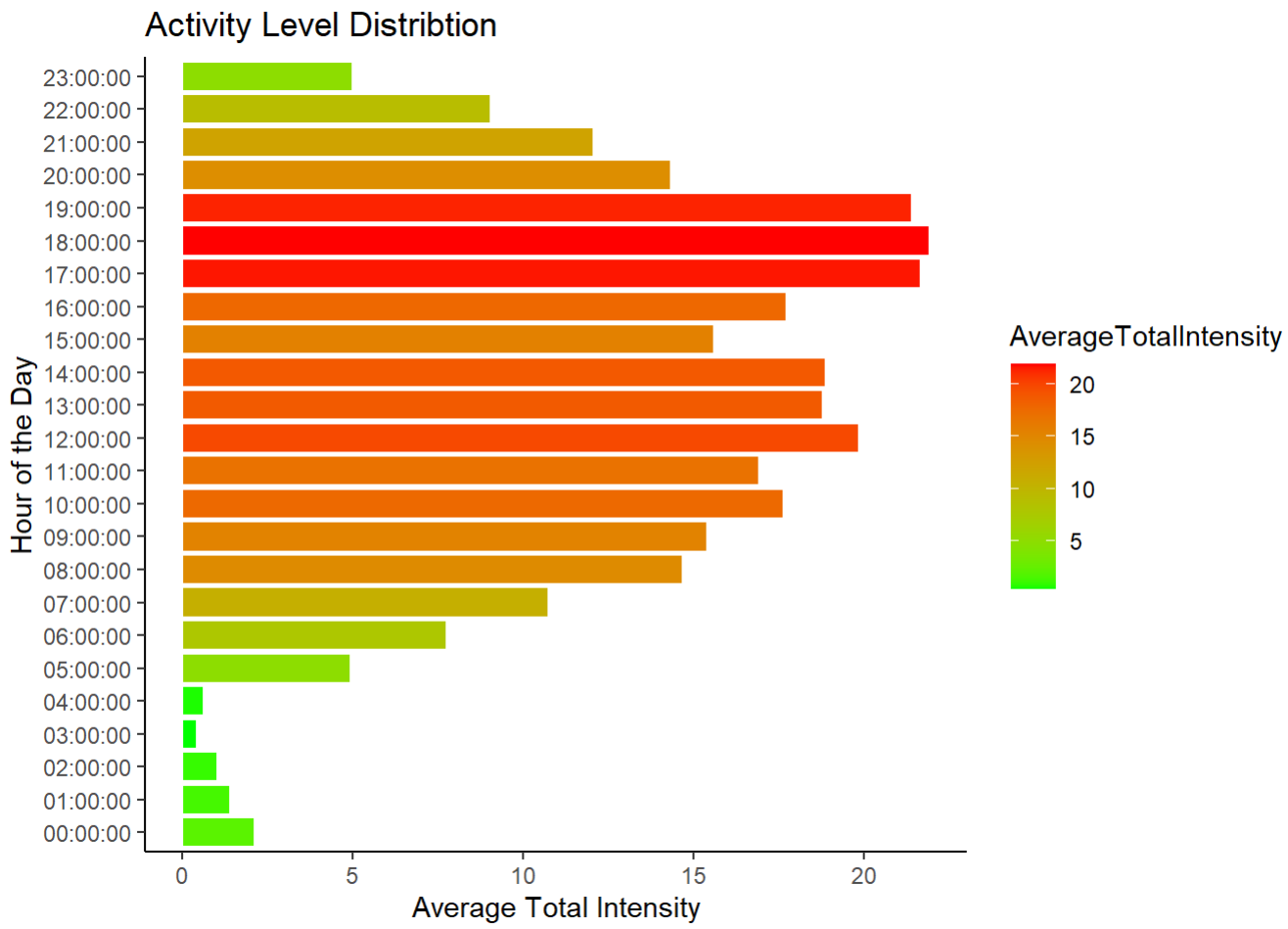
2. Lightly Active - 1 to 3 hour of light activity per day on average
3. Fairly Active - 3 to 5 hour of light activity per day on average
4. Very Active - 5+ hour of light activity per day on average

Based on the pie chart, **52% of users are very active and 30% are fairly active**.

5.2 When are people most active?

```
# Create histogram showing average total intensity for each hour
active_hours <- hourly_activity %>%
  select(Time, TotalIntensity) %>%
  group_by(Time) %>%
  summarise(AverageTotalIntensity = mean(TotalIntensity))

ggplot(active_hours, aes(x=Time, y=AverageTotalIntensity, fill=AverageTotalIntensity)) +
  geom_histogram(stat="identity", color="white") +
  coord_flip() +
  labs(title="Activity Level Distribtion",x="Hour of the Day", y="Average Total Intensity") +
  scale_fill_gradient(low = "green", high = "red") +
  theme_classic()
```

```
## Warning in geom_histogram(stat = "identity", color = "white"): Ignoring unknown
## parameters: `binwidth`, `bins`, and `pad`
```



```
# Find average total intensity each week days
hourly_activity %>%
  select(Id, Datetime, TotalIntensity) %>%
  mutate(Weekday = weekdays(Datetime)) %>%
  group_by(Weekday) %>%
  summarise(AverageTotalIntensity = mean(TotalIntensity))
```

Weekday <chr>	AverageTotalIntensity <dbl>
Friday	12.09309
Monday	12.11220
Saturday	12.90086
Sunday	10.98377
Thursday	11.92690
Tuesday	12.44278
Wednesday	11.75867

7 rows

Based on the histogram, **users are most active between 12pm - 2pm and 5pm - 7pm**. The 12pm - 2pm time frame probably correspond to their work lunch break where they use that time to find lunch. The 5pm - 7pm probably corresponds to after work hours where they are free to exercise or they are commuting back home.

We find user experience their **most intense workout during Saturday while Sunday has the least intensity**. Surprisingly Tuesday has the 2nd highest average intensity despite being in the middle of the week. We believe since users are off work during the weekend they have more time to engage in different activities. Saturday is probably the day people designate to go out, and do thing they enjoy or exercise. Sunday is the designated resting period where people stay home and relax.

5.3 Difference in time between Time Sleep and Time Awake

```
daily_sleep %>%
  mutate(MinutesAwake  = TotalTimeInBed - TotalMinutesAsleep) %>%
  summarise(AverageMinutesAwake = mean(MinutesAwake))
```

AverageMinutesAwake <dbl>
39.30976

1 row

```
daily_sleep %>%
  mutate(MinutesAwake  = TotalTimeInBed - TotalMinutesAsleep) %>%
  group_by(Id) %>%
  summarise(AverageMinutesAwake = mean(MinutesAwake))
```

Id <dbl>	AverageMinutesAwake <dbl>
1503960366	22.920000
1644430081	52.000000
1844505072	309.000000
1927972279	20.800000
2026352035	31.464286
2320127002	8.000000
2347167796	44.533333
3977333714	167.500000
4020332650	30.375000
4319703577	25.307692

1-10 of 24 rows Previous **1** 2 3 Next

Here we calculated the amount of time a user is awake but remain in bed using Total Time in Bed and Total Time Asleep. We found **users on average lie in bed awake for 39.3 minutes**. Excluding 2 abnormal cases, **most user stay in bed for less than an hour** when we inspect each user's average awake in bed times. Upon a quick investigation into the minutes sleep log, we found records that suggest users occasionally either trouble falling asleep, restless nights, trouble getting out of bed or even all 3. Here (https://public.tableau.com/app/profile/kevin.diep/viz/GraphofDailySleepingHabit/Sheet1) is a link to an interactive graph we made on Tableau to check each logs.

5.4 Correlation between sleep time and total intensity for the day

```
# Create plot to check correlation between time spent in bed and total intensity for that day
total_intensity_daily <- hourly_activity %>%
  group_by(Id, Date) %>%
  summarise(DailyIntensity = sum(TotalIntensity))
```

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

```
total_intensity_and_sleep_table <- daily_activity %>%
  drop_na() %>%
  mutate(TotalSleepRecords = as.character(TotalSleepRecords)) %>%
  merge(total_intensity_daily, by=c('Id'='Id', 'Date'='Date'), all.x=TRUE)

ggplot(total_intensity_and_sleep_table, aes(x=DailyIntensity, y=TotalTimeInBed, color=TotalSleep
Records)) +
  geom_point() +
  geom_smooth(method=lm , color="red", se=TRUE) +
  labs(title="Time Spent in Bed After a Day of Activity", x="Daily Intensity", y="Time Spent in
Bed (minutes)") +
  theme_classic()
```

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

The near flat slope indicates there is **little correlation between how intense a person day was and the amount of hour they spend in bed**. Upon further inspection, records show users who sleeps three times a day has notable spent more time in bed for an amount of daily activity than users who slept only once or twice. However, there is little to no difference in records of user was slept once or twice. However, we will note there isn't enough users who slept three times in a day to make any conclusions.

5.5 Quick Observation & Anomalies

```
# List of entries with LoggedActivityDistance greater than 0
daily_activity %>%
  filter(!LoggedActivitiesDistance == 0)
```

Id <dbl>	Date <date>	TotalSteps <int>	TotalDistance <dbl>	TrackerDistance <dbl>	LoggedActivitiesDist
6775888955	2016-04-26	7091	5.27	5.27	1.95
6962181067	2016-04-21	11835	9.71	7.88	4.08
6962181067	2016-04-25	13239	9.27	9.08	2.78
6962181067	2016-05-09	12342	8.72	8.68	3.16
7007744171	2016-04-12	14172	10.29	9.48	4.86
7007744171	2016-04-13	12862	9.65	8.60	4.85
7007744171	2016-04-14	11179	8.24	7.48	3.28
7007744171	2016-04-18	14816	10.98	9.91	4.93
7007744171	2016-04-19	14194	10.48	9.50	4.94
7007744171	2016-04-20	15566	11.31	10.41	4.92

1-10 of 32 rows | 1-6 of 18 columns Previous **1** 2 3 4 Next

We found **LoggedActivityDistance column in daily activity data to be not useful at all**. The TotalDistance column always equal TrackerDistance column except when there is a LoggedActivityDistance columns present. However, when the LoggedActivityDistance column is present, this column combined with the TrackerDistance column, doesn't add up to the TotalDistance column. Given that we don't know what role the LoggedActivityDistance column plays in Total Distance calculation, we will ignore it.

```
# Check entries where users was sedentary for whole day
daily_activity %>%
  select(Id, SedentaryMinutes, Calories) %>%
  filter(SedentaryMinutes == 1440)
```

Id <dbl>	SedentaryMinutes <int>	Calories <int>
1503960366	1440	0
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1348

Id	SedentaryMinutes	Calories
<dbl>	<int>	<int>
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1347

1-10 of 79 rows Previous **1** 2 3 4 5 6 … 8 Next

```
# Check entries where users was sedentary for whole day
daily_activity %>%
  select(Id, SedentaryMinutes, Calories) %>%
  filter(SedentaryMinutes == 1440)
```

Id	SedentaryMinutes	Calories
<dbl>	<int>	<int>
1503960366	1440	0
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1348
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1347
1844505072	1440	1347

1-10 of 79 rows Previous **1** 2 3 4 5 6 … 8 Next

We checked a few entries that indicate a few days where users were completely sedentary. We found some entries indicating 0 Calories burned while other indicate some amount. It should be impossible for 0 calories to be burned despite being sedentary the whole day. Either this was an oversight in the recording program or there is a connection/piece of information we are unaware of.

6. Summary of Analysis

In our initial inspection of the data, we found some entries didn't have sleep data associated with them. Investigating a little further, we found entries without sleep data had an abnormally high recorded Sedentary Minutes in comparison to entries with sleep data. We concluded users without sleep data had their time asleep

recorded as part of Sedentary Minutes. Since there is no way to separate Time spent asleep from Sedimentary Minutes, we can not make any useful conclusion from Sedentary Minutes feature.

When we sum up the activity minute and sleep time, we found roughly 35.7% of entries sum up to either greater than 1440 minutes(24 hours) or less than 1440 minutes(24 hours). While entries less than 24 hour can be explained by users turning off their device, the only explanation for entries over 24 hour is a systematic error somewhere. We found entries without sleeping data add up to 24 hour much more often in comparison to sleeping data. The probably cause of this error lies in the sleeping data. From the Sleep data, we found user on average sleep for roughly 7.7 hours.

Roughly 52%, 30%, 15% and 3% of users are very active, fairly active, lightly active and sedimentary respectively. Although the data doesn't include the participant's gender, we know majority of the participant are at least fairly active people. This a good indicator that the trend we find are applicable to Bellabeat's customer base. We found users are most active from the hour 12pm - 2pm and 5pm - 7pm. A potential cause for this trend is the commute to get lunch and the commute to return home after work. Users are most active on Saturday and least active on Sunday. The weekend provide users the greatest amount of free time. We believe users use Saturday as the day to do thing they enjoy outdoors or exercise. Sunday is the day to stay home and rest.

By comparing time user spend in bed and time asleep, we found users on average stay in bed for 39.3 minutes and most remain in bed for less than one hour. In addition, we found there is no correlation between amount of hour slept and the intensity of that day. There is no noticeable difference between people who slept once per day and people who slept twice per day. There isn't enough entries for people who sleep 3 times a day to make a conclusion.

While checking the data, we found a few sections that doesn't make sense. We found entries with 0 calories burned despite similar data showing some calories burned. Daily Calories Burned should never be 0 regardless of how active a person is each day. We found Logged Activity feature to be a complete enigma. The Logged Distance and Recorded Distance never add up to Total Distance while there is a non zero entry under Logged Distance.

7. Act

To conclude this project, we will our recommendation on how the Bellabeat app can be improved based on our analysis on the FitBit.

1. Some data provided by the FitBit Tracker are either contradictory or does not provide any meaningful use. We suggest implementing a system that ensure incoming data from multiple product doesn't overlap. In addition, the system should be able to tell the difference between when users are asleep and when they simply stationary. Since the self-logged features create contradiction or is not very informative, not including a self logging system would save us time and resources.
2. Fitbit users on average spend about 40 mins in bed while awake. This implies people either have difficulty going to sleep or getting up. We could add a setting in the Bellabeat app to notify users to sleep or to get up if they have been in bed for long period of time.
3. Fitbit users are most active between 12pm - 2pm and 5pm - 7pm. We could have the app give a notification during these time periods to suggest the user go exercise. As an extra feature, we could have the app give healthy food recommendation in the area for them to walk to.
4. We found users are most active during Saturday and least active during Sunday. We suggest having a feature that recommend interesting place to visit to encourage them to walk to said place during Saturday. For Sunday it could recommend healthy ways to relax or some healthy recipes to cook.