Bellabeat Case Study

Google Data Analytics Capstone Project

Shamsur Saikat

March 2022

Introduction

Bellabeat is a high-tech manufacturer of health-focused products for women. The company's products includes a versatile wellness tracker (Leaf), a smart watch (Time) and a smart water bottle (Spring) for hydration tracking. These products connect to the Bellabeat app to report various health and wellness data. Users can also get access to additional perks through Bellabeat membership. In addition to traditional advertising media, the company heavily focuses on digital marketing. The cofounder and CCO, Urška Sršen, wants us to analyze non-Bellabeat smart device usage data to discover insights their usage. These insights will be used to drive company growth through informed marketing strategy.

We will follow the six steps of data analysis process to complete this study - ask, prepare, process, analyze, share and act.

Business Task

Provide high-level recommendations for marketing a Bellabeat product based on analysis of non-Bellabeat smart device usage.

Our key stakeholders are Urška Sršen, Sando Mur (cofounder, Mathematician and executive team member), and the marketing analytics team.

Data Source

For this analysis, we have been encouraged to use FitBit Fitness Tracker Data from Kaggle. This dataset contains personal fitness tracker data from 30 fitbit users who consented to the submission of information about their daily activity, steps, heart rate and sleep monitoring. It was sourced by a third-party, Amazon Mechanical Turk, between 03/12/2016-05/12/2016 and is licensed in the public domain. We are free to use the data for any purpose without restriction of copywright law.

Although this data is comprehensive in terms of information variety and comes from a credible third-party with citation to original source, it has low reliability due to potential sampling bias. This is primarily because of small sample size and unavailability of demographic data. Since our product targets a specific demographic - women - this unreliability should be factored into any business decision derived from this

analysis. Furthermore, at the time of this analysis the collected data is 6 years old. Therefore, the usage behavior may be different due to increased popularity of fitness trackers.

To prepare the data set, it was downloaded and stored locally. We explored the data sources using Excel for initial review and determined which ones to use for our analysis. There are 18 individual data sources (csv files) in long format. We can parse the data by user Id and/or date time stamps. The dataset dailyActivity apparently includes the data from dailyCalories, dailyIntensities and dailySteps. These daily activity attributes also are also broken down by hourly and by minute. For the purpose of this analysis we focused mostly on the daily level. The following data sources were used.

dailyActivity_merged.csv heartrate_seconds_merged.csv hourlyIntensities_merged.csv sleepDay_merged.csv weightLogInfo_merged.csv

Data cleaning and transformation

During exploratory analysis in Excel, we split all datetime columns into separate date and time columns. All date and time columns were then renamed "logDate" and "logTime" and formatted to short date and time format for consistency. The files were then imported to R Studio for further processing and analysis. We'll be using R for ease of documentation using Markdown in addition to having wide variety of analysis and visualization tools.

Installing necessary packages

```
install.packages("readr")
install.packages("tidyverse")
install.packages("dplyr")
install.packages("skimr")
install.packages("janitor")
install.packages("ggplot2")
install.packages("gridExtra")
install.packages("lubridate")


library(readr)
library(tidyverse)
library(dplyr)
library(skimr)
library(janitor)
library(ggplot2)
library(gridExtra)
library(lubridate)
```

Loading CSV files in R

```
# set working directory to folder with all csv data source files
setwd("/cloud/project/Fitbit Data")

dailyActivity <- read_csv("dailyActivity_merged.csv")
dailySleep <- read_csv("sleepDay_merged.csv")
weightLog <- read_csv("weightLogInfo_merged.csv")
hr <- read_csv("heartrate_seconds_merged.csv")
hourlyIntensities <- read_csv("hourlyIntensities_merged.csv")
```

We'll use glimpse() function to view each dataset.

```
glimpse(dailyActivity)
```

```
## Rows: 940
## Columns: 15
## $ Id                       <dbl> 1503960366, 1503960366, 1503960366, 150396036~
## $ logDate                  <chr> "04/12/2016", "04/13/2016", "04/14/2016", "04~
## $ TotalSteps               <dbl> 13162, 10735, 10460, 9762, 12669, 9705, 13019~
## $ TotalDistance            <dbl> 8.50, 6.97, 6.74, 6.28, 8.16, 6.48, 8.59, 9.8~
## $ TrackerDistance          <dbl> 8.50, 6.97, 6.74, 6.28, 8.16, 6.48, 8.59, 9.8~
## $ LoggedActivitiesDistance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ VeryActiveDistance       <dbl> 1.88, 1.57, 2.44, 2.14, 2.71, 3.19, 3.25, 3.5~
## $ ModeratelyActiveDistance <dbl> 0.55, 0.69, 0.40, 1.26, 0.41, 0.78, 0.64, 1.3~
## $ LightActiveDistance      <dbl> 6.06, 4.71, 3.91, 2.83, 5.04, 2.51, 4.71, 5.0~
## $ SedentaryActiveDistance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ VeryActiveMinutes        <dbl> 25, 21, 30, 29, 36, 38, 42, 50, 28, 19, 66, 4~
## $ FairlyActiveMinutes      <dbl> 13, 19, 11, 34, 10, 20, 16, 31, 12, 8, 27, 21~
## $ LightlyActiveMinutes     <dbl> 328, 217, 181, 209, 221, 164, 233, 264, 205, ~
## $ SedentaryMinutes         <dbl> 728, 776, 1218, 726, 773, 539, 1149, 775, 818~
## $ Calories                 <dbl> 1985, 1797, 1776, 1745, 1863, 1728, 1921, 203~
```

```
glimpse(hourlyIntensities)
```

```
## Rows: 22,099
## Columns: 5
## $ Id               <dbl> 1503960366, 1503960366, 1503960366, 1503960366, 15039~
## $ logDate          <chr> "04/12/2016", "04/12/2016", "04/12/2016", "04/12/2016~
## $ logTime          <time> 00:00:00, 01:00:00, 02:00:00, 03:00:00, 04:00:00, 05~
## $ TotalIntensity   <dbl> 20, 8, 7, 0, 0, 0, 0, 0, 13, 30, 29, 12, 11, 6, 36, 5~
## $ AverageIntensity <dbl> 0.333333, 0.133333, 0.116667, 0.000000, 0.000000, 0.0~
```

```
glimpse(hr)
```

```
## Rows: 1,048,575
## Columns: 4
## $ Id      <dbl> 2022484408, 2022484408, 2022484408, 2022484408, 2022484408, 20~
## $ logDate <chr> "04/12/2016", "04/12/2016", "04/12/2016", "04/12/2016", "04/12~
## $ logTime <time> 07:21:00, 07:21:05, 07:21:10, 07:21:20, 07:21:25, 07:22:05, 0~
## $ Value   <dbl> 97, 102, 105, 103, 101, 95, 91, 93, 94, 93, 92, 89, 83, 61, 60~
```

```
glimpse(dailySleep)
```

```
## Rows: 413
## Columns: 5
## $ Id                 <dbl> 1503960366, 1503960366, 1503960366, 1503960366, 150~
## $ logDate            <chr> "04/12/2016", "04/13/2016", "04/15/2016", "04/16/20~
## $ TotalSleepRecords  <dbl> 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ TotalMinutesAsleep <dbl> 327, 384, 412, 340, 700, 304, 360, 325, 361, 430, 2~
## $ TotalTimeInBed     <dbl> 346, 407, 442, 367, 712, 320, 377, 364, 384, 449, 3~
```

```
glimpse(weightLog)
```

```
## Rows: 67
## Columns: 9
## $ Id             <dbl> 1503960366, 1503960366, 1927972279, 2873212765, 2873212~
## $ logDate        <chr> "05/02/2016", "05/03/2016", "04/13/2016", "04/21/2016",~
## $ logTime        <time> 23:59:59, 23:59:59, 01:08:52, 23:59:59, 23:59:59, 23:5~
## $ WeightKg       <dbl> 52.6, 52.6, 133.5, 56.7, 57.3, 72.4, 72.3, 69.7, 70.3, ~
```

```
## $ WeightPounds   <dbl> 115.9631, 115.9631, 294.3171, 125.0021, 126.3249, 159.6~
## $ Fat            <dbl> 22, NA, NA, NA, NA, 25, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ BMI            <dbl> 22.65, 22.65, 47.54, 21.45, 21.69, 27.45, 27.38, 27.25,~
## $ IsManualReport <lgl> TRUE, TRUE, FALSE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, ~
## $ LogId          <dbl> 1.46223e+12, 1.46232e+12, 1.46051e+12, 1.46128e+12, 1.4~
```

We'll add some columns to the daily activity data and hourly intensities to aid in our analysis. From the date column, we can extract the day of the week.

```
dailyActivity$weekDay <- strptime(dailyActivity$logDate,format="%m/%d/%Y") %>%
  as.Date(dailyActivity$logDate) %>%
  weekdays(., abbreviate = FALSE)

dailyActivity$weekDayNum <- strptime(dailyActivity$logDate,format="%m/%d/%Y") %>%
  as.Date(dailyActivity$logDate) %>%
  wday(., week_start=1)

hourlyIntensities$weekDay <- strptime(hourlyIntensities$logDate,format="%m/%d/%Y") %>%
  as.Date(hourlyIntensities$logDate) %>%
  weekdays(., abbreviate = FALSE)

hourlyIntensities$weekDayNum <- strptime(hourlyIntensities$logDate,format="%m/%d/%Y") %>%
  as.Date(hourlyIntensities$logDate) %>%
  wday(., week_start=1)
```

Analysis

Below is a list of insights drawn from our analysis of the data sets. Details of the analysis are available in the individual sections (click to navigate).

1. User categories - Sleep tracking and weight/bmi monitoring feature is underutilized. 88% of users were regular about activity tracking compared to 50% in sleep tracking and 0% for weight tracking.
2. Daily Steps - 79% users don't meet the recommended daily steps of 10,000. The average is 8239 steps.
3. Activeness - People spend about 81% of their time sedentary. 48.5% people do not meet weekly recommended level of moderate and vigorous activity.
4. Activeness- Users are most active between 5-7pm on weekdays and on Saturday afternoon. They are least active on Sundays.
5. Calories Burned - Vigorous activity is the strongest contributing factor to active calories burned, followed by total steps.
6. Sleep Analysis - Users get 7 hours of sleep on average, showing a decline in sleep duration with more sedentary hours.
7. Heart Rate - Increased activity can contribute to better heart health by reducing resting heart rate.
8. Weight and BMI - Users with higher step count have lower bmi in the healthy range.
9. Weight and BMI - 61% of weight data recorded were manual making tracking inconvenient.

User categories In order to understand consumer behavior, we need to look at how much the devices are being used and for what purpose. Grouping the datasets dailyActivity, dailySleep and weightLog, we can find the distinct number of users, the frequency of usage (days logged), and the distribution.

```r
#Grouping daily activity for daily averages of users
dailyActivity_UserSummary <- dailyActivity %>%
 group_by(Id) %>%
 summarise(n = n(), steps = round(mean(TotalSteps)), calories = round(mean(Calories)),
           VeryActiveMinutes = round(mean(VeryActiveMinutes)),
           FairlyActiveMinutes = round(mean(FairlyActiveMinutes)),
           LightlyActiveMinutes = round(mean(LightlyActiveMinutes)),
           SedentaryMinutes = round(mean(SedentaryMinutes)))

#grouping sleep log by user
dailySleep_UserSummary <- dailySleep %>%
  group_by(Id) %>%
  summarise(n=n(), averageTimeAsleep = mean(TotalMinutesAsleep), averageTimeInBed = mean(TotalTimeInBed

#grouping weight log by user
weightLog_UserSummary <- weightLog %>%
  group_by(Id) %>%
  summarise(n=n(), weight_lbs = mean(WeightPounds), max_weight = max(WeightPounds), min_weight = min(We

n_distinct(dailyActivity_UserSummary$Id)
```

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

```r
n_distinct(dailySleep_UserSummary$Id)
```

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

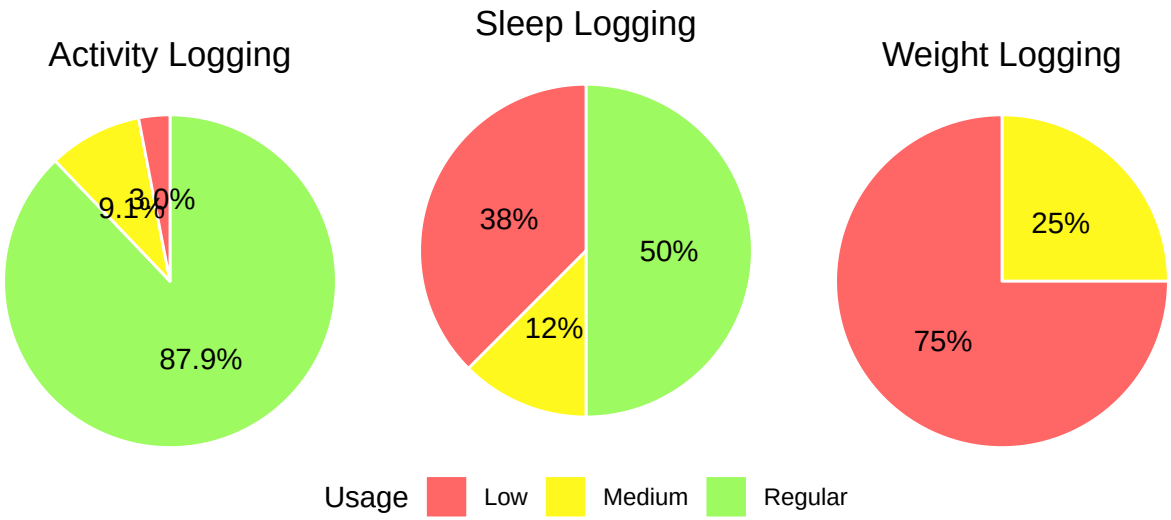
```r
n_distinct(weightLog_UserSummary$Id)
```

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

We have 33, 24 and 8 users in the activity, sleep and weight log datasets, respectively. Let's categorize these users based on number of days logged -"Rare", "Moderate" and "Regular" for usage of 0-10, 11-20, 21-31 days. Then we group the data based on these categories to aggregate total number and percentage of users in each category. Finally, we visualize the distribution by usage for all three types of health information.

```r
#Assigning categories to users based on number of days logged
dailyActivity_UserSummary$Usage <- ifelse(dailyActivity_UserSummary$n<=10,"Low",
                ifelse(dailyActivity_UserSummary$n>10 & dailyActivity_UserSummary$n<21, "Medium",
                       "Regular"))

dailySleep_UserSummary$Usage <- ifelse(dailySleep_UserSummary$n<=10,"Low",
                ifelse(dailySleep_UserSummary$n>10 & dailySleep_UserSummary$n<21, "Medium", "Regular"))

weightLog_UserSummary$Usage <- ifelse(weightLog_UserSummary$n<=10,"Low",
                ifelse(weightLog_UserSummary$n>10 & weightLog_UserSummary$n<21, "Medium", "Regular"))


#Grouping users based on number of days logged
dailyActivityUsage <- dailyActivity_UserSummary %>%
  group_by(Usage) %>%
  summarise(userCount=n(), average_usage= round(mean(n),0))

dailySleepUsage <- dailySleep_UserSummary %>%
  group_by(Usage) %>%
  summarise(userCount=n(), average_usage= round(mean(n),0))
```

```r
weightLogUsage <- weightLog_UserSummary %>%
  group_by(Usage) %>%
  summarise(userCount=n(), average_usage= round(mean(n),0))

#Sorting categories based on no. of users in each category
dailyActivityUsage <- dailyActivityUsage[order(-dailyActivityUsage$average_usage),]
dailySleepUsage <- dailySleepUsage[order(-dailySleepUsage$average_usage),]
weightLogUsage <- weightLogUsage[order(-weightLogUsage$average_usage),]

#Adding percentage column for no. of users in each category
dailyActivityUsage$perc <-scales::percent(
                          dailyActivityUsage$userCount/sum(dailyActivityUsage$userCount))
dailySleepUsage$perc <- scales::percent(dailySleepUsage$userCount/sum(dailySleepUsage$userCount))
weightLogUsage$perc <- scales::percent(weightLogUsage$userCount/sum(weightLogUsage$userCount))


#Visual representation of usage
gg_activity <- ggplot(dailyActivityUsage, aes(x = "", y = userCount, fill = Usage)) +
     geom_col(color = "white") +
     geom_text(aes(label = perc),
               position = position_stack(vjust = 0.5)) +
     coord_polar(theta = "y") +
     scale_fill_manual(values = c("#ff6666", "#fff81f", "#9efa61")) +
     labs(title="Activity Logging") +
     theme_void()+
     theme(legend.position="none", plot.title = element_text(hjust = 0.5))

gg_usageSleep <- ggplot(dailySleepUsage, aes(x = "", y = userCount, fill = Usage)) +
     geom_col(color = "white") +
     geom_text(aes(label = perc),
               position = position_stack(vjust = 0.5)) +
     coord_polar(theta = "y") +
     scale_fill_manual(values = c("#ff6666", "#fff81f", "#9efa61")) +
     labs(title="Sleep Logging") +
     theme_void()+
     theme(legend.position="bottom", plot.title = element_text(hjust = 0.5))

gg_usageWeight <- ggplot(weightLogUsage, aes(x = "", y = userCount, fill = Usage)) +
     geom_col(color = "white") +
     geom_text(aes(label = perc),
               position = position_stack(vjust = 0.5)) +
     coord_polar(theta = "y") +
     scale_fill_manual(values = c("#ff6666", "#fff81f", "#9efa61")) +
     labs(title="Weight Logging") +
     theme_void()+
     theme(legend.position="none", plot.title = element_text(hjust = 0.5))


grid.arrange(gg_activity, gg_usageSleep,gg_usageWeight, ncol=3)
```



It looks like 87.9% of our users are regular when it comes to dailyActivity recording with the device. We have 9.1% moderate and 3% low frequency users. If I look at the same statistic for sleep recording and weight logging we see a different picture. The distribution also changes drastically. We have only 50% users tracking sleep regularly and 25% users logging weight.

So far we have seen that dailyActivity tracking is what users are mostly utilizing the device for. I'd say the sleep tracking and weight/bmi monitoring feature is underutilized. We'll have to do further analysis on all of the datasets to get more insight of their usage behavior.

Daily Activity Summary We will start with dailyActivity and check the summary of key parameters.

```
dailyActivity %>%
  select(TotalSteps, TotalDistance, SedentaryMinutes, LightlyActiveMinutes, FairlyActiveMinutes, VeryAc
  summary()
```

```
##    TotalSteps    TotalDistance    SedentaryMinutes LightlyActiveMinutes
##  Min.   :    0   Min.   : 0.000   Min.   :   0.0   Min.   :  0.0
##  1st Qu.: 3790   1st Qu.: 2.620   1st Qu.: 729.8   1st Qu.:127.0
##  Median : 7406   Median : 5.245   Median :1057.5   Median :199.0
##  Mean   : 7638   Mean   : 5.490   Mean   : 991.2   Mean   :192.8
##  3rd Qu.:10727   3rd Qu.: 7.713   3rd Qu.:1229.5   3rd Qu.:264.0
##  Max.   :36019   Max.   :28.030   Max.   :1440.0   Max.   :518.0
##  FairlyActiveMinutes VeryActiveMinutes    Calories
##  Min.   :  0.00      Min.   :  0.00    Min.   :   0
##  1st Qu.:  0.00      1st Qu.:  0.00    1st Qu.:1828
##  Median :  6.00      Median :  4.00    Median :2134
##  Mean   : 13.56      Mean   : 21.16    Mean   :2304
##  3rd Qu.: 19.00      3rd Qu.: 32.00    3rd Qu.:2793
##  Max.   :143.00      Max.   :210.00    Max.   :4900
```

From summary of sedentary minutes, Max value is 1440 minutes, which is entire duration of the day. We'll make an assumption here that the device registers days when unworn as sedentary by default. This can bias our findings. Therefore, we filtered out rows with sedentary minutes 1440 where total steps value is also 0 (as a secondary check).

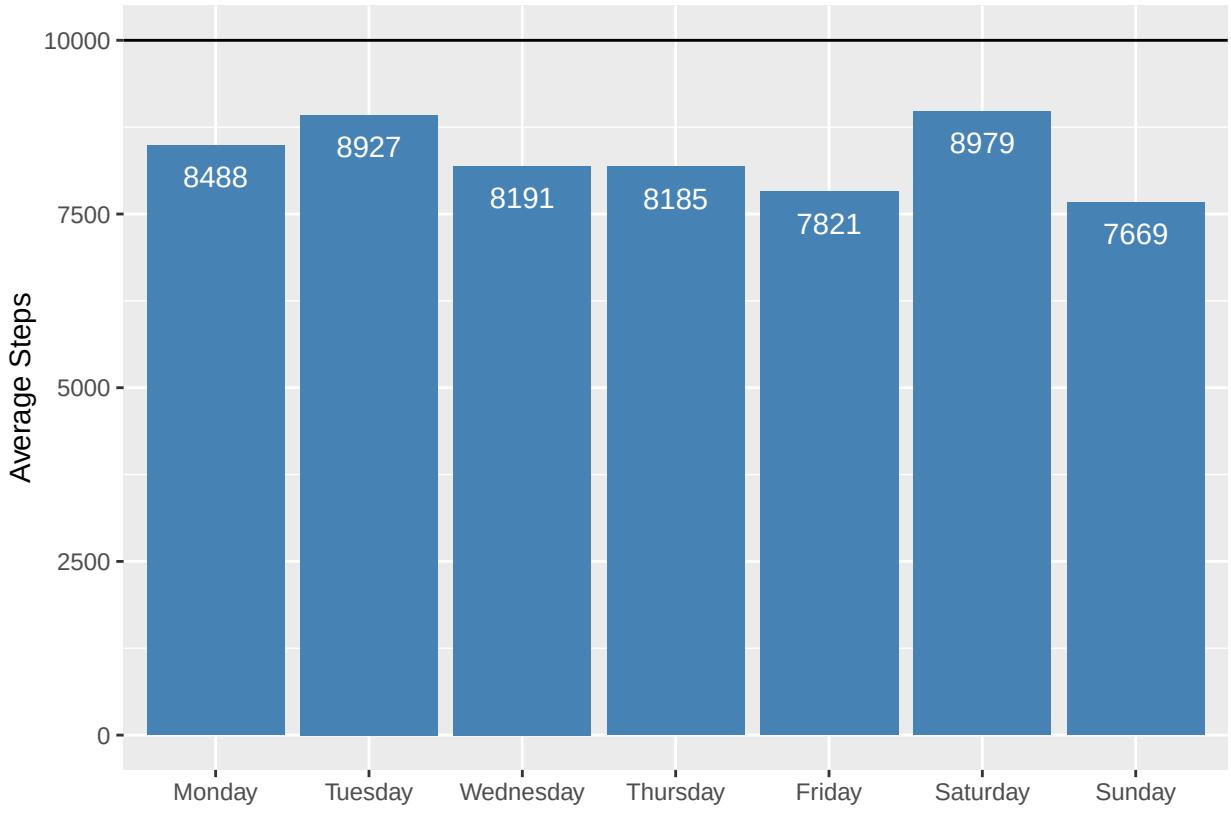
```
# Removing rows of data with no activity
dailyActivity <- dailyActivity %>%
  filter(SedentaryMinutes != 1440 & TotalSteps !=0)
```

Daily Steps Average steps taken by users every day is 8239, which is below the recommended 10,000 by CDC. In fact, 79% users average below this recommended level. Users get the highest amount of steps on Saturdays, and the least on Sundays. Throughout the week their step count generally decreases. To ensure users get their target steps we can notify users to walk more at specific times during the day to complete the goal number of steps.

```
#calculating percentage of users who average less than 10,000 steps a day
round(nrow(dailyActivity_UserSummary[dailyActivity_UserSummary$steps < 10000,]) / nrow(dailyActivity_Us
```

```
## [1] 79
```

```
gg_stepsWeekday <- dailyActivity %>%
  group_by(weekDay) %>%
  summarise(steps = mean(TotalSteps), weekDayNum = mean(weekDayNum)) %>%
  ggplot(., aes(x=reorder(weekDay, weekDayNum), y=steps)) +
  geom_bar(stat = "identity", fill="steelblue")  +
  geom_hline(yintercept = 10000) +
  labs(x="", y = "Average Steps") +
  geom_text(aes(label = round(steps)), vjust = 2, colour = "white")

print(gg_stepsWeekday)
```

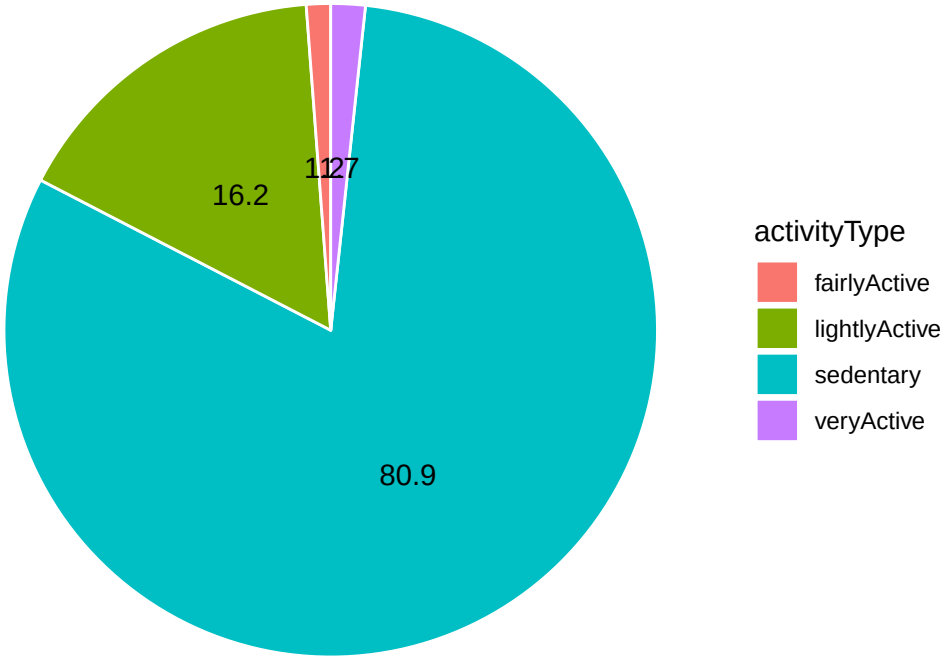


Activeness We can get an idea of how active our users are in general by looking at the distribution of time between sedentary, lightly active, fairly active and very active minutes. We total the different categories of active minutes, add a new column 'totalActiveMinutes' in dailyActivity dataset and corresponding summary

column in dailyActivity_UserSummary. Then we'll represent time spent in each category as percentage of total activity.

```
#Totaling minutes across different activity levels
dailyActivity <- dailyActivity %>%
  mutate(totalActiveMinutes = VeryActiveMinutes + FairlyActiveMinutes +
                                  LightlyActiveMinutes + SedentaryMinutes)

dailyActivity_UserSummary <- dailyActivity_UserSummary %>%
  mutate(average_totalActiveMinutes = SedentaryMinutes + LightlyActiveMinutes +
                                      FairlyActiveMinutes + VeryActiveMinutes)

dailyActivityRatio <-dailyActivity_UserSummary %>%
  summarise(sedentary=mean(SedentaryMinutes/average_totalActiveMinutes),
            lightlyActive=mean(LightlyActiveMinutes/average_totalActiveMinutes),
            fairlyActive=mean(FairlyActiveMinutes/average_totalActiveMinutes),
            veryActive=mean(VeryActiveMinutes/average_totalActiveMinutes)) %>%
  summarise(sedentary = round(sedentary*100,1), lightlyActive = round(lightlyActive*100,1), fairlyActiv

dailyActivityRatio <- gather(dailyActivityRatio, activityType, percDay)

gg_ActivityRatio <- ggplot(dailyActivityRatio, aes(x = "", y = percDay, fill = activityType)) +
     geom_col(color = "white") +
     geom_text(aes(label = percDay),
               position = position_stack(vjust = 0.5)) +
     coord_polar(theta = "y") +
     labs(title="Daily activity ratio") +
     theme_void()

print(gg_ActivityRatio)
```

Daily activity ratio



Looks like the average user spends 80.9% of their time sedentary, 16.2% lightly active, 1.2% fairly active and 1.7% very active. This indicates a high level of sedentary time so it is worthwhile to investigate if they meet the recommended amount of activity.

According to CDC guideline, adults should aim for 150 minutes of moderate activity or 75 minutes of vigorous activity per week. It can also be a combination of the two spread across multiple days. This translates to approximately 21 minutes of moderate and 11 minutes of vigorous activity daily. Assuming that moderate and vigorous activity is equivalent to fairly active and very active, we can see what fraction of the users meet this criteria.

About 51.5% of users meet the recommended activity level, which leaves 48.5% of users vulnerable to health complication due to lack of physical activity. ***It is possible to market to these users by allowing our product to track their progress through the week and notify them daily of remainder of their weekly goal.***

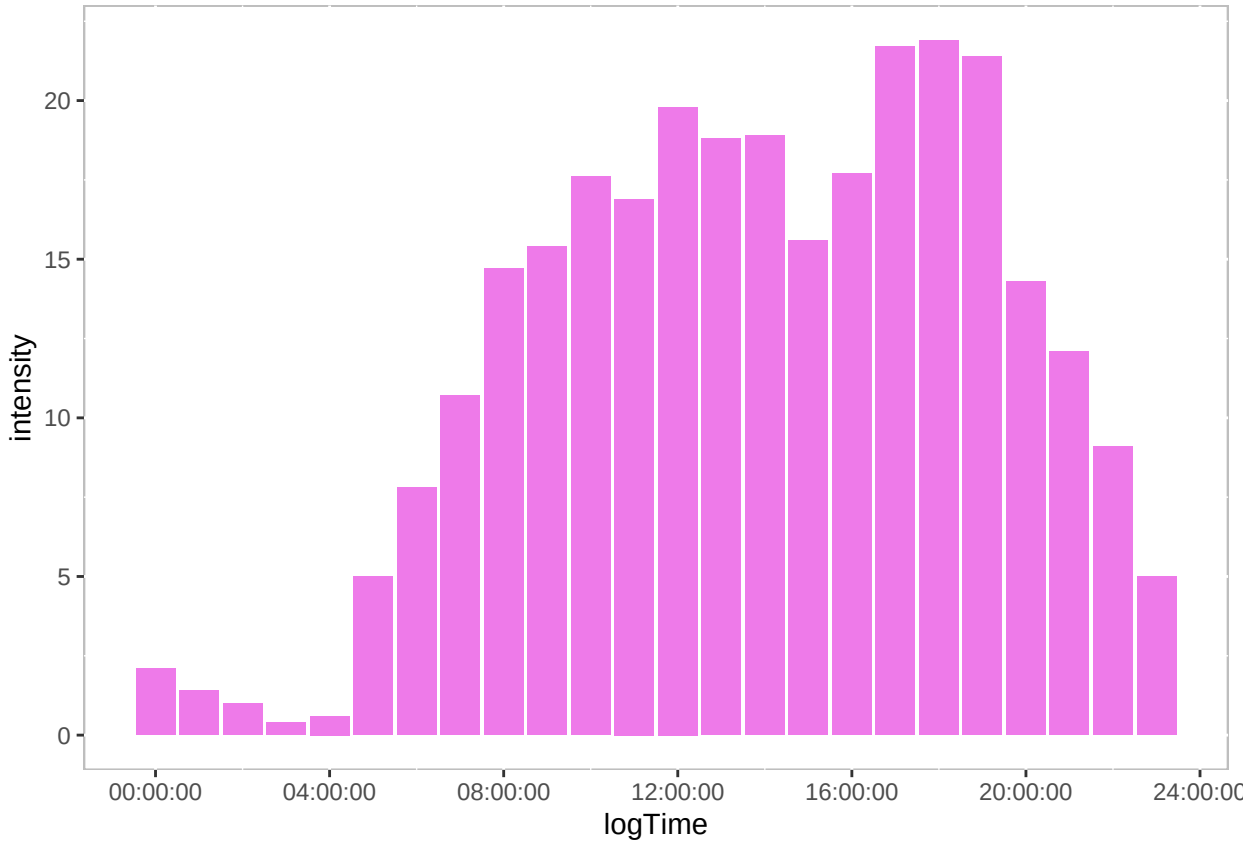
```
nrow(dailyActivity_UserSummary[(dailyActivity_UserSummary$VeryActiveMinutes > 11 | dailyActivity_UserSu
```

```
## [1] 51.51515
```

In addition to the extent of activeness, we can find out when they are active. We group the HourlyIntensities data by weekday and calculate the average of TotalIntensity for every hour. From the column chart below, we see that users are most active between 5pm and 7pm. We can also visualize the activeness intensity over days of the week throughout the day in the heatmap below. Users tend to be active for less duration over the weekend compared to the weekday. On weekdays users are most active in the evening between 5pm to 7pm and on weekends it's Saturday afternoon.

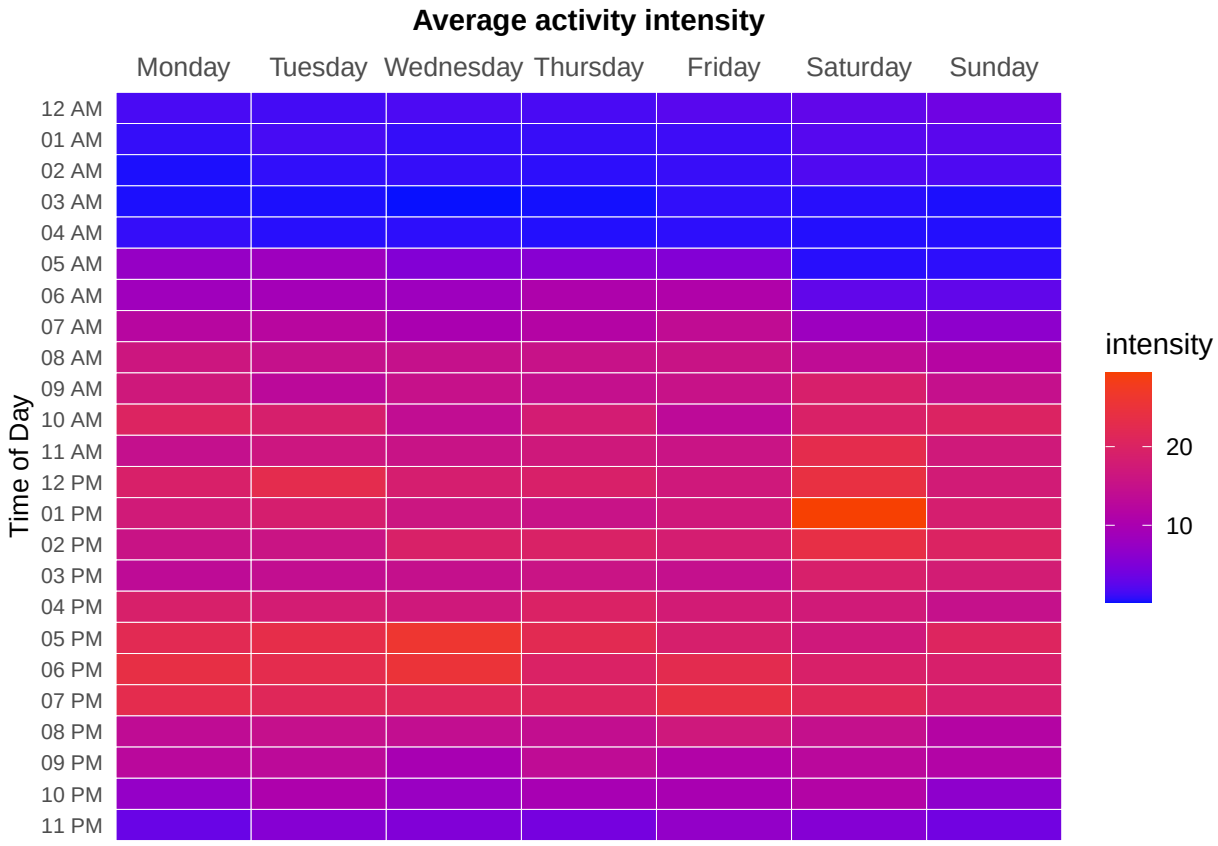
```
#plotting intensity over time of day
gg_hourlyIntensity <-  hourlyIntensities %>%
  group_by(logTime) %>%
  summarise(intensity = round(mean(TotalIntensity),1)) %>%
  ggplot(., aes(x=logTime, y=intensity)) +
  geom_bar(stat = "identity", fill="orchid2") +
```

```
  theme(panel.background = element_rect(fill = 'white', color = 'grey'))

print(gg_hourlyIntensity)
```




```
#Grouping intensity by weekday and time
intensityWeekDay <- hourlyIntensities %>%
  group_by(weekDay, logTime) %>%
  summarise(intensity = round(mean(TotalIntensity),1), weekDayNum = mean(weekDayNum), .groups = "keep")

#Converting time from hh:mm:ss to h am/pm
intensityWeekDay$logTimeTransformed <- format(strptime(intensityWeekDay$logTime, format='%H'), '%I %p')

# Create a theme that is a bit more friendly. This isn't required to do the visualization,
# but it's a painful heatmap to look at otherwise. We start with a minimalist theme... and
# then basically make it even more minimalist. Thank you, Tufte and Few.
theme_heatmap <- theme_light() +                  # Start with a minimalist theme
  theme(panel.grid = element_blank(),             # Remove the gridlines
        panel.border = element_blank(),           # Remove the border around the heatmap
        plot.title = element_text(face = "bold", # Make the title bold
                                  size = 11,      # Adjust the title size
                                  hjust = 0.5),   # Center the title
        axis.ticks = element_blank(),             # Remove the axis tickmarks
        axis.title.x = element_blank(),           # Turn off the x-axis title
        axis.title.y = element_text(size=10),     # Adjust the size of the y-axis title
        axis.text.y = element_text(size = 8),     # Adjust the size of the y-axis labels
        axis.text.x = element_text(size = 10),)    # Adjust the size of the x-axis labels
        #legend.position = "none")                 # Turn off the legend
```

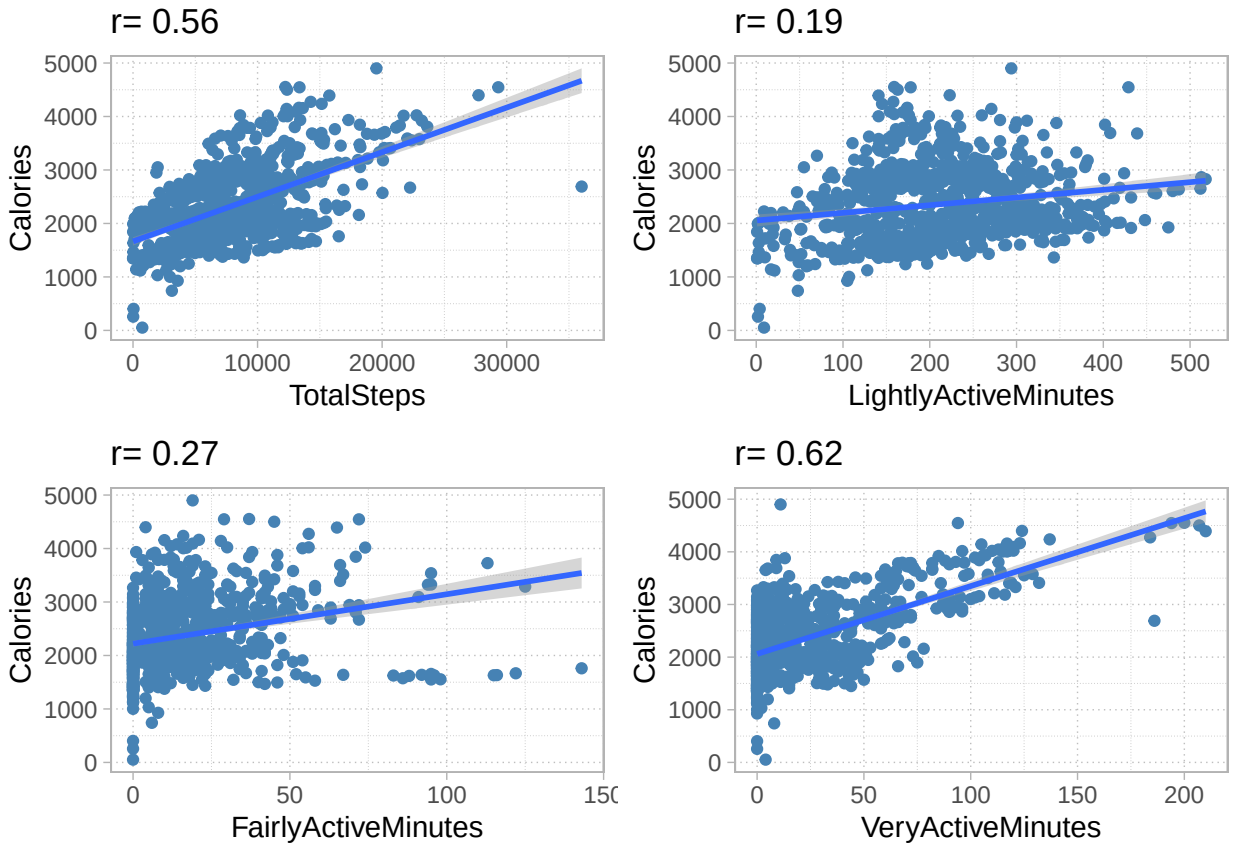
```r
# Create the plot.

gg_activityIntensity <- ggplot(intensityWeekDay, mapping = aes(x = reorder(weekDay, weekDayNum),
                      y = reorder(logTimeTransformed, desc(logTime)), fill = intensity)) +
  geom_tile(colour="white") +  # This makes the heatmap (the colour is the lines in the grid)
  scale_fill_gradient(low = "#0810ff", high="#f74002") +  # The colour range to use
  scale_x_discrete(expand = c(0,0), position = "top") +
  labs(title = "Average activity intensity", y = "Time of Day") +
  theme_heatmap  # Apply the theme defined earlier for styling


print(gg_activityIntensity)
```



Calories Burned On average, users burned 2304 calories daily. The actual amount of calories burned can be dependent on multiple other factors such as age, gender and body mass. We expect to see correlation between activity level and calories burned. From this dataset we can find how the different types of activity compare in regards to their relationship with calories burned. Plotting the total steps, lightly active, fairly active and very active minutes show that users can burn more calories by increasing any of these throughout their day. However, very active minutes and total steps had strongest correlation (0.62 and 0.56 respectively). This information can help users monitor these factors to achieve their goals and be more active in general.

We can also incentivize users towards our product for their weight goals. A person's weight gained or lost is a direct result of net calories. In order to maintain a healthy weight, users can set daily calorie goals that our device can help them track and notify when unmet.

```r
#creating custom theme for scatter plots
theme_customScatter <-theme_light() + theme(panel.background = element_rect(fill = 'white', color = 'gr
        panel.grid.major = element_line(color = 'grey', linetype = 'dotted'),
        panel.grid.minor = element_line(color = 'grey', linetype = 'dotted'))

#calculating correlation coefficients between calories burned and the different activity types
cor_Calories1 <- round(cor(dailyActivity$TotalSteps, dailyActivity$Calories),2)
cor_Calories2 <- round(cor(dailyActivity$LightlyActiveMinutes, dailyActivity$Calories),2)
cor_Calories3 <- round(cor(dailyActivity$FairlyActiveMinutes, dailyActivity$Calories),2)
cor_Calories4 <- round(cor(dailyActivity$VeryActiveMinutes, dailyActivity$Calories),2)

#creating scatter plots
gg_Calories1 <- ggplot(dailyActivity, aes(TotalSteps, Calories)) +
                geom_point(color="steelblue") + geom_smooth(formula = y ~ x,method=lm) +
                labs(title = paste("r=",cor_Calories1 )) +
                theme_customScatter

gg_Calories2 <- ggplot(dailyActivity, aes(LightlyActiveMinutes, Calories)) +
                geom_point(color="steelblue") + geom_smooth(formula = y ~ x,method=lm) +
                labs(title = paste("r=",cor_Calories2 )) +
                theme_customScatter

gg_Calories3 <- ggplot(dailyActivity, aes(FairlyActiveMinutes, Calories)) +
                geom_point(color="steelblue") + geom_smooth(formula = y ~ x,method=lm) +
                labs(title = paste("r=",cor_Calories3 )) +
                theme_customScatter

gg_Calories4 <- ggplot(dailyActivity, aes(VeryActiveMinutes, Calories)) +
                geom_point(color="steelblue") + geom_smooth(formula = y ~ x,method=lm) +
                labs(title = paste("r=",cor_Calories4 )) +
                theme_customScatter

grid.arrange(gg_Calories1, gg_Calories2, gg_Calories3, gg_Calories4, ncol=2)
```



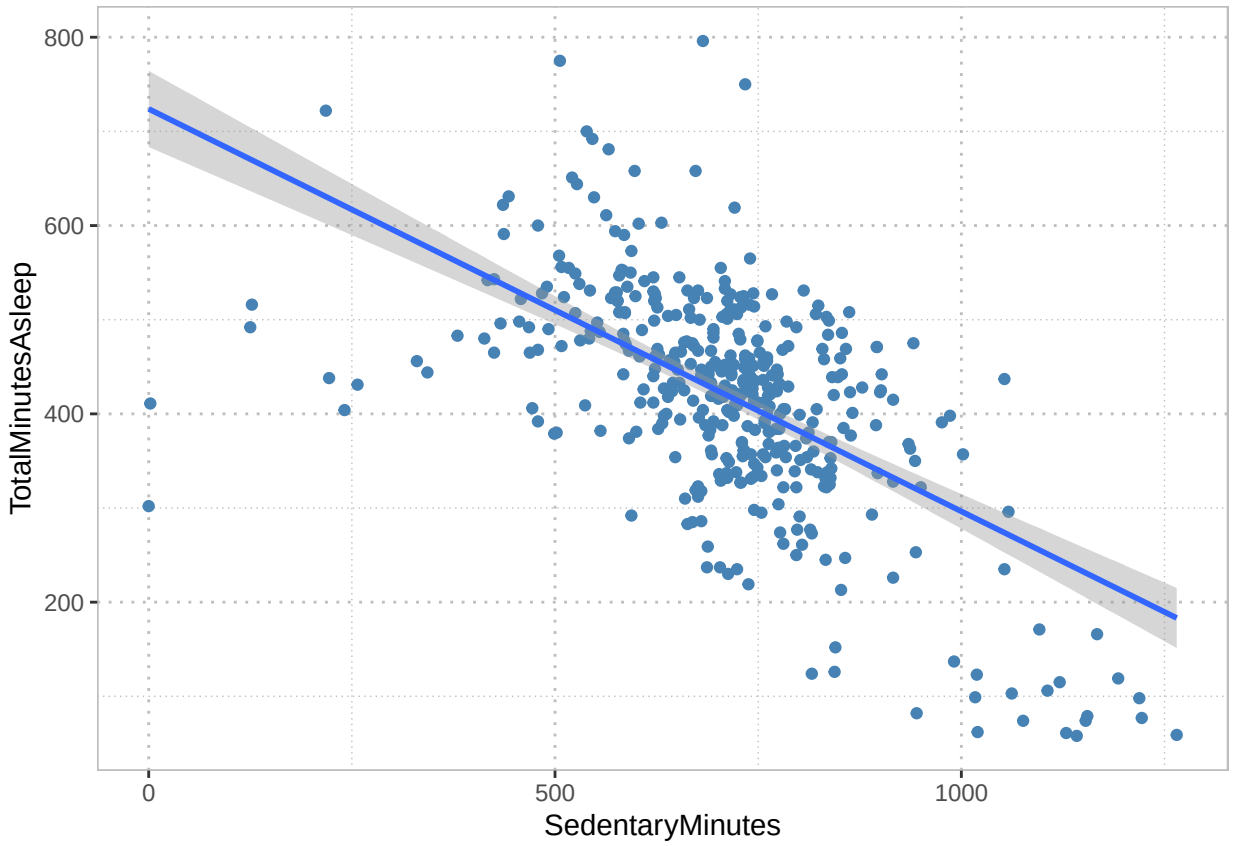
Sleep Analysis An important component of health is rest, primarily sleep. When it comes to sleep, the quality of sleep one gets is just as important as the amount. However, our dataset only has duration in bed, duration asleep and sleep by minute. By using summary() function on the dailySleep dataset, we see that users average 7 hrs of sleep and 0.6 hrs or 36 minutes of awake time in bed. By merging the daily activity dataset with sleep we can explore if there is any relationship between activity and sleep. We can see a negative correlation between sedentary minutes and time asleep. Users who are less active also average less sleep.

```
#Calculating sleep efficiency based on proportion of time asleep and time in bed
dailySleep <- dailySleep %>%
  mutate(sleepEfficiency = round((TotalMinutesAsleep)*100/TotalTimeInBed,1))

#Calculating averages of sleep data
dailySleep %>%
  select(TotalMinutesAsleep, TotalTimeInBed, sleepEfficiency) %>%
  summarise(timeAsleep = round(mean(TotalMinutesAsleep)/60,1), timeInBed= round(mean(TotalTimeInBed)/60
```

```
## # A tibble: 1 x 3
##   timeAsleep timeInBed sleepEfficiency
##        <dbl>     <dbl>           <dbl>
## 1          7       7.6            91.7
```

```
#merging the activity and sleep data sets
dailyActivityMerged <- merge(dailyActivity, dailySleep, by = c("Id", "logDate"), all=TRUE)
dailyActivityMerged <- dailyActivityMerged %>%
  mutate(timeInBedAwake = TotalTimeInBed - TotalMinutesAsleep)
```

```
gg_sleep <- dailyActivityMerged %>%
  drop_na(TotalMinutesAsleep) %>%
  ggplot(., aes(x=SedentaryMinutes, y=TotalMinutesAsleep)) +
  geom_point(color="steelblue") + geom_smooth(formula = y ~ x,method=lm) +
  theme(panel.background = element_rect(fill = 'white', color = 'grey'),
        panel.grid.major = element_line(color = 'grey', linetype = 'dotted'),
        panel.grid.minor = element_line(color = 'grey', linetype = 'dotted'))

print(gg_sleep)
```



Heart Rate Heart rate for adults can vary based on multiple factors such as age, gender, fitness level, stress, etc. A normal range is considered between 60-100 beats per minute (bpm). A quick summary of our heart rate data set shows 7 participants report average heart rate within normal range.

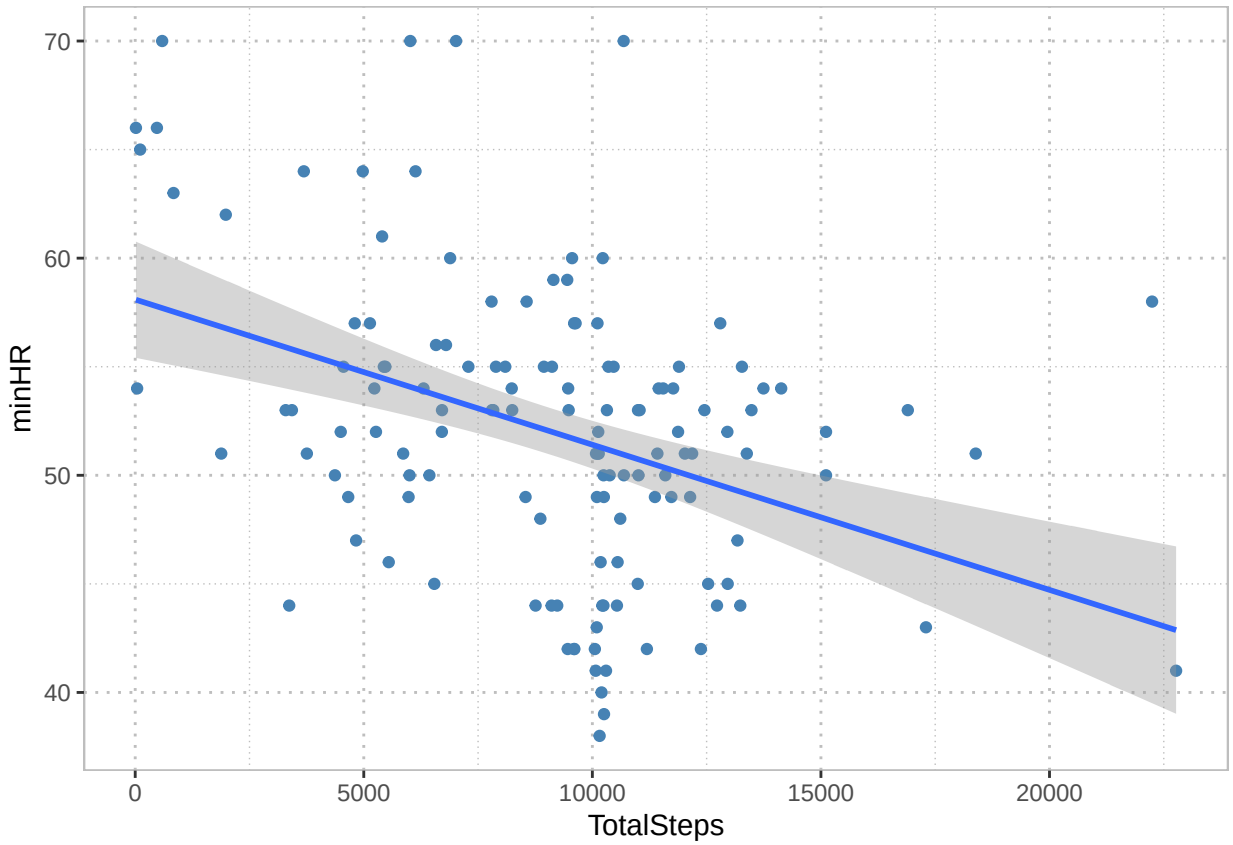
```
#heart rate summary by user
hr %>%
  group_by(Id) %>%
  summarise(minHR = min(Value), maxHR = max(Value), avgHR = mean(Value))
```

```
## # A tibble: 7 x 4
##           Id minHR maxHR avgHR
##        <dbl> <dbl> <dbl> <dbl>
## 1 2022484408    38   203  80.2
## 2 2026352035    63   125  93.8
## 3 2347167796    49   195  76.7
```

```
## 4 4020332650    46   191  82.3
## 5 4388161847    39   180  66.1
## 6 4558609924    44   199  81.7
## 7 5553957443    47   106  62.8
```

For an individual user, it is beneficial to know several heart rate related parameters, such as resting heart rate, maximum heart rate, or heart rate during exercising. Despite the small sample size, we checked for trend between average daily steps and resting heart rate (a general indicator of cardiac health and fitness). We took the minimum heart rate from each day for every user. Then we merged this data frame with dailyActivity summary by user Id, and plotted average steps over min heart rate. We can see from the trend that users who average higher step count tend to have lower resting heart rate (as expected).

```
## Scouting for trend between heart rate, activity and sleep

#extracting resting heart rate as daily lowest heart rates for every user
dailyRestingHr <- hr %>%
  group_by(Id, logDate) %>%
  summarise(minHR = min(Value), maxHR = max(Value), avgHR = mean(Value))

#merging resting hr with daily activity
dailyActivityMerged <- merge(dailyActivityMerged, dailyRestingHr, by=c("Id", "logDate"), all=TRUE)

#plotting the resting hr over daily step removing rows with empty hr data
gg_heartrate <- dailyActivityMerged %>%
  drop_na(minHR) %>%
  ggplot(., aes(x=TotalSteps, y=minHR)) +
  geom_point(color = "steelblue") + geom_smooth(formula = y ~ x,method=lm) +
  theme(panel.background = element_rect(fill = 'white', color = 'grey'),
        panel.grid.major = element_line(color = 'grey', linetype = 'dotted'),
        panel.grid.minor = element_line(color = 'grey', linetype = 'dotted'))

print(gg_heartrate)
```



While the small sample size leaves our findings inconclusive, we can use this ability to analyze hr on individual level over time to monitor heart health. In addition to long term benefit, we can provide users with real time hr monitoring during daily exercise to categorize their activity level as fat burning, cardio or peak. This allows users to increase or decrease the intensity of their exercise according to their goals.

Weight and BMI Body weight can be very useful indicator of one's physical health. Particularly, BMI - which accounts for height - can be a screening tool for over or underweight. According to CDC, a healthy BMI range is between 18.5 to 24.5. Among the 8 users in our dataset, 37.5% have a healthy weight and 62.5% are in the overweight or obese range.

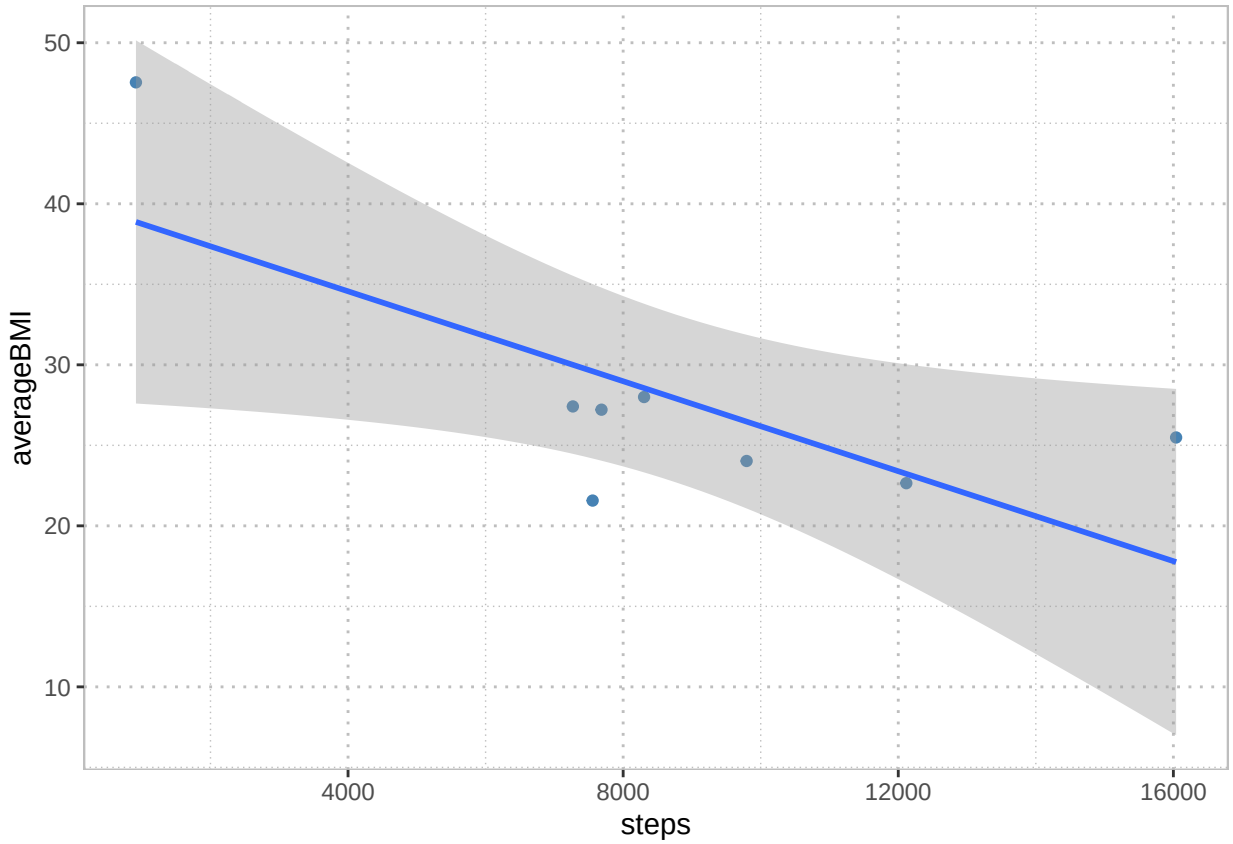
```
weightLog_UserSummary$BMIcategory <- ifelse(weightLog_UserSummary$averageBMI<18.5,"Underweight",
                ifelse(weightLog_UserSummary$averageBMI>=18.5 & weightLog_UserSummary$averageBMI<=24.5,

weightLog_UserSummary %>%
  group_by(BMIcategory) %>%
  summarise(percBMIcategory = n()*100/nrow(.))
```

```
## # A tibble: 3 x 2
##   BMIcategory percBMIcategory
##   <chr>                 <dbl>
## 1 Healthy                37.5
## 2 Obese                  12.5
## 3 Overweight             50
```

We see a trend if we merge the weightSummary dataset with dailyActivity summary and plot average BMI against average steps. Users who average higher number of steps tend to have lower BMI in the healthy range. While this fits the expected narrative, this is not conclusive of the overall population since we have very few

number of users. We need more data points to be confident in our conclusion. Additionally, a healthy weight is not just a factor of steps taken or calories burned but also calories consumed. Without a holistic analysis, causal relationships cannot be determined.

```
gg_bmi <- merge(weightLog_UserSummary, dailyActivity_UserSummary[c("Id", "steps")], by="Id", all=FALSE)
  ggplot(., aes(x=steps, y=averageBMI)) +
  geom_point(color="steelblue") + geom_smooth(formula = y ~ x,method=lm) +
    theme(panel.background = element_rect(fill = 'white', color = 'grey'),
        panel.grid.major = element_line(color = 'grey', linetype = 'dotted'),
        panel.grid.minor = element_line(color = 'grey', linetype = 'dotted'))

print(gg_bmi)
```



Earlier in our analysis, we saw that very few users actually log their weights. 61% of the weight log was manual, which may be a deterrent to logging weight regularly. Manual logging can be tedious for users and they may also not remember to log weight daily/frequently. Having pairing functionality with smart scales can be one weigh of encouraging users to be more regular with weight tracking.

```
weightLog %>%
 group_by(IsManualReport) %>%
 summarise(n=n()) %>%
  mutate(perc = scales::percent(n/sum(n)))
```

```
## # A tibble: 2 x 3
##   IsManualReport     n perc
##   <lgl>          <int> <chr>
## 1 FALSE             26 39%
## 2 TRUE              41 61%
```

Recommendations

1. Enable users to set daily/weekly goals of action items such as steps, exercise, sleep and calories burned (specific to weight goals).
2. Notify users of progress and remainder of their daily and weekly goals. For example, if an user is behind on their calorie goal by the end of workday, send a push notification suggesting a walk in units of distance or time equivalent to the calorie goal deficit.
3. Alert users of long periods of inactivity and send reminders to move. Alert users of bedtime based on their settings.
4. Ensure adequate battery life and comfortable form factor to encourage sleep tracking and reporting metrics like sleep duration, bed time consistency and sleep efficiency/latency.
5. Report weekly trends of important cardiovascular metrics like resting heart rate, max heart rate and heart rate recovery post exercise. Allow heart rate tracking during physical activity for users to be able to control intensity of exercise and maintain heart rate in target zone for fat burn, cardio, etc.
6. Allow pairing or data sync with smart weighing scales for more regular weight tracking.
7. "Gamify" health tracking through social interactions of the app by allowing users to share their activity and participate in daily/weekly challenges together. Enable access to health educational content.