Bellabeat Case Study

Publisher: Jackie Lo

Last Update: 2021-07-28

Bellabeat Marketing Stragey/Analysis

Project Introduction

This project contains one of two case studies of Google's Data Analytics Professional Certification provided by Google. The requirements for the case study are for the analyst to do data analysis using FitBit Fitness Tracker data to provide high-level marketing strategy recommendations for Bellabeat through the process of Ask, Prepare, Process, Analyst, Share, and Act process.

Bellabeat is a high-tech company that manufactures health-focused innovative products like an app, Leaf (bracelet), Time (watch), and Spring (water bottle). The company also provides subscription services or membership programs for users to have 24/7 access to their fully personalized guidance on nutrition, activity, sleep, health and beauty, and mindfulness-based on their lifestyle and goals.

Objective

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

Import library needed:

```
#install.packages("tidyverse")
#install.packages("ids")
#install.packages("gridExtra")
library(tidyverse)
library(lubridate)
library(skimr)
library(janitor)
library(ids)
library(gridExtra)
```

Importing/Read .csv datasets into dataframes:

```
daily_activity = read.csv("~/Case_Study_bellabeat/Fitabase Data 4.12.16-5.12.16/dailyActivity_merged.csv
heart_rate_seconds = read.csv("~/Case_Study_bellabeat/Fitabase Data 4.12.16-5.12.16/heartrate_seconds_m
sleep_day = read.csv("~/Case_Study_bellabeat/Fitabase Data 4.12.16-5.12.16/sleepDay_merged.csv")
weight_log = read.csv("~/Case_Study_bellabeat/Fitabase Data 4.12.16-5.12.16/weightLogInfo_merged.csv")
fitness_trend = read.csv("~/Case_Study_bellabeat/25.csv")
```

```
activity_log = read.csv("~/Case_Study_bellabeat/ACTIVITY/ACTIVITY_1599810432505.csv")
sleep_log = read.csv("~/Case_Study_bellabeat/SLEEP/SLEEP_1599810433552.csv")
heartrate_auto_log = read.csv("~/Case_Study_bellabeat/HEARTRATE_AUTO/HEARTRATE_AUTO_1599810433761.csv")
```

Prepare and Process

Heart Rate Dataset

```
#checking data quality of the original
head(heart_rate_seconds)
```

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

```
skim_without_charts(heart_rate_seconds)
```

Table 1: Data summary

Name	heart_rate_seconds
Number of rows	2483658
Number of columns	3
Column type frequency:	
character	1
numeric	2
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Time	0	1	19	21	0	961274	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
Id	0	1	5.513765e+09	1950223761.2	2022484408	3388161845	7553957446	8962181067	8877689391
Value	0	1	7.733000e+01	19.4	36	63	73	88	203

```
anyNA(heart_rate_seconds)
```

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

```
anyDuplicated(heart_rate_seconds)
```

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

```
#cleaning/confirming data for quality to be place in a dataset that will be used for analysis
heart_rate_sec = heart_rate_seconds %>%
  clean_names() %>%
  distinct() %>%
  mutate(time = parse_date_time(time, "%m/%d/%Y %I:%M:%S %p")) %>%
  separate(col = time,
           into = c("date", "time"), sep = " ") %>%
  mutate(date = as.Date(date),
         time = format(strptime(time, "%H:%M:%S"), "%H")) %>%
  rename(heart_rate = value) %>%
  group_by(id, date, time) %>%
  summarize(heart_rate = mean(heart_rate)) %>%
  arrange(id, date, time)
```

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

```
#final quality check
head(heart_rate_sec)
```

```
## # A tibble: 6 x 4
## # Groups:   id, date [1]
##           id date       time  heart_rate
##        <dbl> <date>     <chr>      <dbl>
## 1 2022484408 2016-04-12 07          83.2
## 2 2022484408 2016-04-12 08          68.6
## 3 2022484408 2016-04-12 09          66.4
## 4 2022484408 2016-04-12 10         107.
## 5 2022484408 2016-04-12 11          67.8
## 6 2022484408 2016-04-12 12          66.2
```

```
anyNA(heart_rate_sec)
```

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

```
anyDuplicated(heart_rate_sec)
```

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

```
#checking the second heart rate data for quality of the original
head(heartrate_auto_log)
```

```
##      ï..date  time heartRate
## 1 2019-09-13 06:53        80
## 2 2019-09-13 07:23        65
## 3 2019-09-13 09:53        51
## 4 2019-09-13 10:53        87
## 5 2019-09-13 11:23        60
## 6 2019-09-13 12:23        56
```

```
skim_without_charts(heartrate_auto_log)
```

Table 4: Data summary

Name	heartrate_auto_log
Number of rows	2430
Number of columns	3
Column type frequency:	
character	2
numeric	1
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ï..date	0	1	10	10	0	91	0
time	0	1	5	5	0	1023	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
heartRate	0	1	68.61	20.92	47	54	62	77	150

```
anyNA(heartrate_auto_log)
```

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

```
anyDuplicated(heartrate_auto_log)
```

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

```
#random id generator
set.seed(2430)
activity_id = sort(sample.int(2430,2430))

#cleaning/confirming data for quality to be place in a dataset that will be used for analysis
```

```
heartrate_auto = heartrate_auto_log %>%
  clean_names() %>%
  distinct() %>%
  rename(date = i_date) %>%
  mutate(date = as.Date(date, "%Y-%m-%d"),
         time = substr(time, 1, 5),
         id = activity_id) %>%
  arrange(id, date, time)

#final quality check
head(heartrate_auto)
```

```
##         date  time heart_rate id
## 1 2019-09-13 06:53         80  1
## 2 2019-09-13 07:23         65  2
## 3 2019-09-13 09:53         51  3
## 4 2019-09-13 10:53         87  4
## 5 2019-09-13 11:23         60  5
## 6 2019-09-13 12:23         56  6
```

```
anyNA(heartrate_auto)
```

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

```
anyDuplicated(heartrate_auto)
```

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

Sleep Dataset

```
#checking data quality of the original
head(sleep_day)
```

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

```
skim_without_charts(sleep_day)
```

Table 7: Data summary

Name	sleep_day
Number of rows	413
Number of columns	5
Column type frequency:	
character	1
numeric	4
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
SleepDay	0	1	20	21	0	31	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
Id	0	1	5.000979e+09	2.906036e+09	1503960366	3977333714	4702921684	6962181067	8792009665
TotalSleepRecords	0	1	1.120000e+00	8.050000e-01	1	1	1	1	3
TotalMinutesAsleep	0	1	4.194700e+02	1.218340e+02	58	361	433	490	796
TotalTimeInBed	0	1	4.586400e+02	1.227100e+02	61	403	463	526	961

```
anyNA(sleep_day)
```

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

```
anyDuplicated(sleep_day)
```

```
## [1] 162
```

```
#cleaning/confirming data for quality to be place in a dataset that will be used for analysis
sleep_d = sleep_day %>%
  distinct() %>%
  clean_names() %>%
  separate(col = sleep_day,
           into = c("sleep_day", "sleep_time"), sep = " ") %>%
  mutate(sleep_day = as.Date(sleep_day,"%m/%d/%Y")) %>%
  rename(date = sleep_day) %>%
  mutate(time_awake = total_time_in_bed - total_minutes_asleep) %>%
  arrange(id,date) %>%
  select(-total_sleep_records, -sleep_time)
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 410 rows [1, 2, 3, 4,
## 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
#final quality check
head(sleep_d)
```

```
##           id       date total_minutes_asleep total_time_in_bed time_awake
## 1 1503960366 2016-04-12                  327               346         19
## 2 1503960366 2016-04-13                  384               407         23
## 3 1503960366 2016-04-15                  412               442         30
## 4 1503960366 2016-04-16                  340               367         27
## 5 1503960366 2016-04-17                  700               712         12
## 6 1503960366 2016-04-19                  304               320         16
```

```
anyNA(sleep_d)
```

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

```
anyDuplicated(sleep_d)
```

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

```
#checking the second sleep data for quality of the original
head(sleep_log)
```

```
##      ï..date lastSyncTime deepSleepTime shallowSleepTime wakeTime      start
## 1 2018-09-29   1538285362             0                0        0 1538245800
## 2 2018-09-30   1538396903           141              253        2 1538255880
## 3 2018-10-01   1539148718             0                0        0 1538418600
## 4 2018-10-02   1539148718            80               49        0 1538418180
## 5 2018-10-03   1539148718             0                0        0 1538591400
## 6 2018-10-04   1539148718             0                0        0 1538677800
##         stop
## 1 1538245800
## 2 1538279640
## 3 1538418600
## 4 1538425920
## 5 1538591400
## 6 1538677800
```

```
skim_without_charts(sleep_log)
```

Table 10: Data summary

Name	sleep_log
Number of rows	538
Number of columns	7

Column type frequency:	
character	1
numeric	6
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ï..date	0	1	10	10	0	269	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
lastSyncTime	0	1	1.558860e+09	2263513.76	1653828536	1550407001	1565672299	1569240513	1574058669
deepSleepTime	0	1	1.559000e+01	76.66	0	0	0	0	827
shallowSleepTime	0	1	2.371000e+01	94.97	0	0	0	0	878
wakeTime	0	1	1.000000e-02	0.12	0	0	0	0	2
start	0	1	1.557395e+09	2180772.36	1653824580	1554619460	1562351400	1568313000	1574015400
stop	0	1	1.557397e+09	2181880.96	1653824580	1554619460	1562351400	1568313000	1574058660

```
anyNA(sleep_log)
```

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

```
anyDuplicated(sleep_log)
```

```
## [1] 270
```

```
#random id generator
set.seed(269)
sleep_id = sort(sample.int(269,269))

#cleaning/confirming data for quality to be place in a dataset that will be used for analysis
sleep_l = sleep_log %>%
  distinct() %>%
  clean_names() %>%
  rename(date = i_date,
         time_awake = wake_time) %>%
  mutate(date = as.Date(date, "%Y-%m-%d"), id = sleep_id,
         total_minutes_asleep = deep_sleep_time + shallow_sleep_time,
         total_time_in_bed = total_minutes_asleep + time_awake) %>%
  select(id, date, total_minutes_asleep,
         total_time_in_bed, time_awake) %>%
  arrange(id, date)

#final quality check
head(sleep_l)
```

```
##   id       date total_minutes_asleep total_time_in_bed time_awake
## 1  1 2018-09-29                    0                 0          0
## 2  2 2018-09-30                  394               396          2
## 3  3 2018-10-01                    0                 0          0
## 4  4 2018-10-02                  129               129          0
## 5  5 2018-10-03                    0                 0          0
## 6  6 2018-10-04                    0                 0          0
```

```
anyNA(sleep_l)
```

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

```
anyDuplicated(sleep_l)
```

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

Weight Log Dataset

```
#checking data quality of the original
head(weight_log)
```

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

```
anyNA(weight_log)
```

```
## [1] TRUE
```

```
anyDuplicated(weight_log)
```

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

```
#cleaning/confirming data for quality to be place in a dataset that will be used for analysis
weight_l = weight_log %>%
  distinct() %>%
  clean_names() %>%
```

```
  mutate(date = as.Date(date, "%m/%d/%Y"), height = sqrt(weight_kg/bmi)*100) %>%
  select(-weight_pounds, -log_id, -is_manual_report, -fat) %>%
  arrange(id, date)

#final quality check
head(weight_l)
```

```
##           id       date weight_kg   bmi   height
## 1 1503960366 2016-05-02      52.6 22.65 152.3908
## 2 1503960366 2016-05-03      52.6 22.65 152.3908
## 3 1927972279 2016-04-13     133.5 47.54 167.5757
## 4 2873212765 2016-04-21      56.7 21.45 162.5840
## 5 2873212765 2016-05-12      57.3 21.69 162.5352
## 6 4319703577 2016-04-17      72.4 27.45 162.4045
```

```
anyNA(weight_l)
```

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

```
anyDuplicated(weight_l)
```

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

Fitness Trend Dataset

```
#checking data quality of the original
head(fitness_trend)
```

```
##         date step_count mood calories_burned hours_of_sleep bool_of_active
## 1 2017-10-06       5464  200             181              5               0
## 2 2017-10-07       6041  100             197              8               0
## 3 2017-10-08         25  100               0              5               0
## 4 2017-10-09       5461  100             174              4               0
## 5 2017-10-10       6915  200             223              5             500
## 6 2017-10-11       4545  100             149              6               0
##   weight_kg
## 1        66
## 2        66
## 3        66
## 4        66
## 5        66
## 6        66
```

```
anyNA(fitness_trend)
```

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

```
anyDuplicated(fitness_trend)
```

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

```
#random id generator
set.seed(96)
fitness_id = sort(sample.int(96,96))

#cleaning/confirming data for quality to be place in a dataset that will be used for analysis
fitness_t = fitness_trend %>%
  distinct() %>%
  clean_names() %>%
  rename(total_minutes_asleep = hours_of_sleep, total_steps = step_count) %>%
  mutate(id = fitness_id,
         date = as.Date(date, "%Y-%m-%d"),
         total_minutes_asleep = total_minutes_asleep * 60) %>%
  select(-mood, -bool_of_active) %>%
  arrange(id, date)

fitness_sleep_trend = fitness_t %>%
  select(id, date, total_minutes_asleep)

fitness_weight_trend = fitness_t %>%
  select(id, date, weight_kg)

fitness_activity_trend = fitness_t %>%
  select(id, date, total_steps, calories_burned)

#final quality check
head(fitness_t)
```

```
##         date total_steps calories_burned total_minutes_asleep weight_kg id
## 1 2017-10-06        5464             181                  300        66  1
## 2 2017-10-07        6041             197                  480        66  2
## 3 2017-10-08          25               0                  300        66  3
## 4 2017-10-09        5461             174                  240        66  4
## 5 2017-10-10        6915             223                  300        66  5
## 6 2017-10-11        4545             149                  360        66  6
```

```
head(fitness_activity_trend)
```

```
##   id       date total_steps calories_burned
## 1  1 2017-10-06        5464             181
## 2  2 2017-10-07        6041             197
## 3  3 2017-10-08          25               0
## 4  4 2017-10-09        5461             174
## 5  5 2017-10-10        6915             223
## 6  6 2017-10-11        4545             149
```

```
head(fitness_sleep_trend)
```

```
##   id       date total_minutes_asleep
## 1  1 2017-10-06                 300
## 2  2 2017-10-07                 480
## 3  3 2017-10-08                 300
## 4  4 2017-10-09                 240
## 5  5 2017-10-10                 300
## 6  6 2017-10-11                 360
```

```
head(fitness_weight_trend)
```

```
##   id       date weight_kg
## 1  1 2017-10-06        66
## 2  2 2017-10-07        66
## 3  3 2017-10-08        66
## 4  4 2017-10-09        66
## 5  5 2017-10-10        66
## 6  6 2017-10-11        66
```

```
anyNA(fitness_t)
```

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

```
anyNA(fitness_activity_trend)
```

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

```
anyNA(fitness_sleep_trend)
```

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

```
anyNA(fitness_weight_trend)
```

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

```
anyDuplicated(fitness_t)
```

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

```
anyDuplicated(fitness_activity_trend)
```

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

```
anyDuplicated(fitness_sleep_trend)
```

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

```
anyDuplicated(fitness_weight_trend)
```

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

Activities Dataset

```
#checking data quality of the original
head(daily_activity)
```

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

```
anyNA(daily_activity)
```

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

```
anyDuplicated(daily_activity)
```

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

```
#cleaning/confirming data for quality to be place in a dataset that will be used for analysis
activity_d = daily_activity %>%
  distinct() %>%
```

```
  clean_names() %>%
  rename(date = activity_date, calories_burned = calories) %>%
  mutate(date = as.Date(date, "%m/%d/%Y")) %>%
  #mutate(avg_active_min = fairly_active_minutes + lightly_active_minutes + sedentary_minutes +
                              #very_active_minutes) %>%
  select(-tracker_distance, -logged_activities_distance, -moderately_active_distance,
         -light_active_distance, -sedentary_active_distance,
         -fairly_active_minutes, -lightly_active_minutes, -sedentary_minutes, -very_active_minutes) %>%
  arrange(id, date)

#final quality check
head(activity_d)
```

```
##           id       date total_steps total_distance very_active_distance
## 1 1503960366 2016-04-12       13162           8.50                 1.88
## 2 1503960366 2016-04-13       10735           6.97                 1.57
## 3 1503960366 2016-04-14       10460           6.74                 2.44
## 4 1503960366 2016-04-15        9762           6.28                 2.14
## 5 1503960366 2016-04-16       12669           8.16                 2.71
## 6 1503960366 2016-04-17        9705           6.48                 3.19
##   calories_burned
## 1            1985
## 2            1797
## 3            1776
## 4            1745
## 5            1863
## 6            1728
```

```
anyNA(activity_d)
```

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

```
anyDuplicated(activity_d)
```

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

```
#checking the second Activity data for quality of the original
head(activity_log)
```

```
##      ï..date lastSyncTime steps distance runDistance calories
## 1 2018-09-29   1538285362  8017     5341          65      157
## 2 2018-09-30   1538396903  4002     2717         127       86
## 3 2018-10-01   1539148718  2379     1484         123       47
## 4 2018-10-02   1539148718     0        0           0        0
## 5 2018-10-03   1539148718  8051     5501         182      165
## 6 2018-10-04   1539148718  6504     4443         195      136
```

```
anyNA(activity_log)
```

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

```
anyDuplicated(activity_log)
```

```
## [1] 270
```

```
#generating random id number
set.seed(269)
activity_id = sort(sample.int(269, 269))

#cleaning/confirming data for quality to be place in a dataset that will be used for analysis
activity_l = activity_log %>%
  distinct() %>%
  clean_names() %>%
  rename(date = i_date, calories_burned = calories, very_active_distance = run_distance,
         total_steps = steps, total_distance = distance) %>%
  mutate(date = as.Date(date, "%Y-%m-%d"),
         total_distance = total_distance / 1000,
         very_active_distance = very_active_distance / 1000,
         id = c(activity_id)) %>%
  select(-last_sync_time) %>%
  arrange(id, date)

#final quality check
head(activity_l)
```

```
##         date total_steps total_distance very_active_distance calories_burned id
## 1 2018-09-29        8017          5.341                 0.065             157  1
## 2 2018-09-30        4002          2.717                 0.127              86  2
## 3 2018-10-01        2379          1.484                 0.123              47  3
## 4 2018-10-02           0          0.000                 0.000               0  4
## 5 2018-10-03        8051          5.501                 0.182             165  5
## 6 2018-10-04        6504          4.443                 0.195             136  6
```

```
anyNA(activity_l)
```

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

```
anyDuplicated(activity_l)
```

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

Analyze & Share

```
#binding all the activity dataset
activity = bind_rows(activity_d, activity_l, fitness_activity_trend)
sleep = bind_rows(sleep_d, sleep_l, fitness_sleep_trend)
heartrate = rbind(heart_rate_sec, heartrate_auto)
weight = bind_rows(fitness_weight_trend, weight_l)
```

```
paste("The number of unique IDs in Activity dataset =",n_unique(activity$id))
```

```
## [1] "The number of unique IDs in Activity dataset = 302"
```

```
paste("The number of unique IDs in Sleep dataset =", n_unique(sleep$id))
```

```
## [1] "The number of unique IDs in Sleep dataset = 293"
```

```
paste("The number of unique IDs in Heartrate dataset =", n_unique(heartrate$id))
```

```
## [1] "The number of unique IDs in Heartrate dataset = 2444"
```

```
paste("The number of unique IDs in Weight dataset =", n_unique(weight$id))
```

```
## [1] "The number of unique IDs in Weight dataset = 104"
```

Visualization

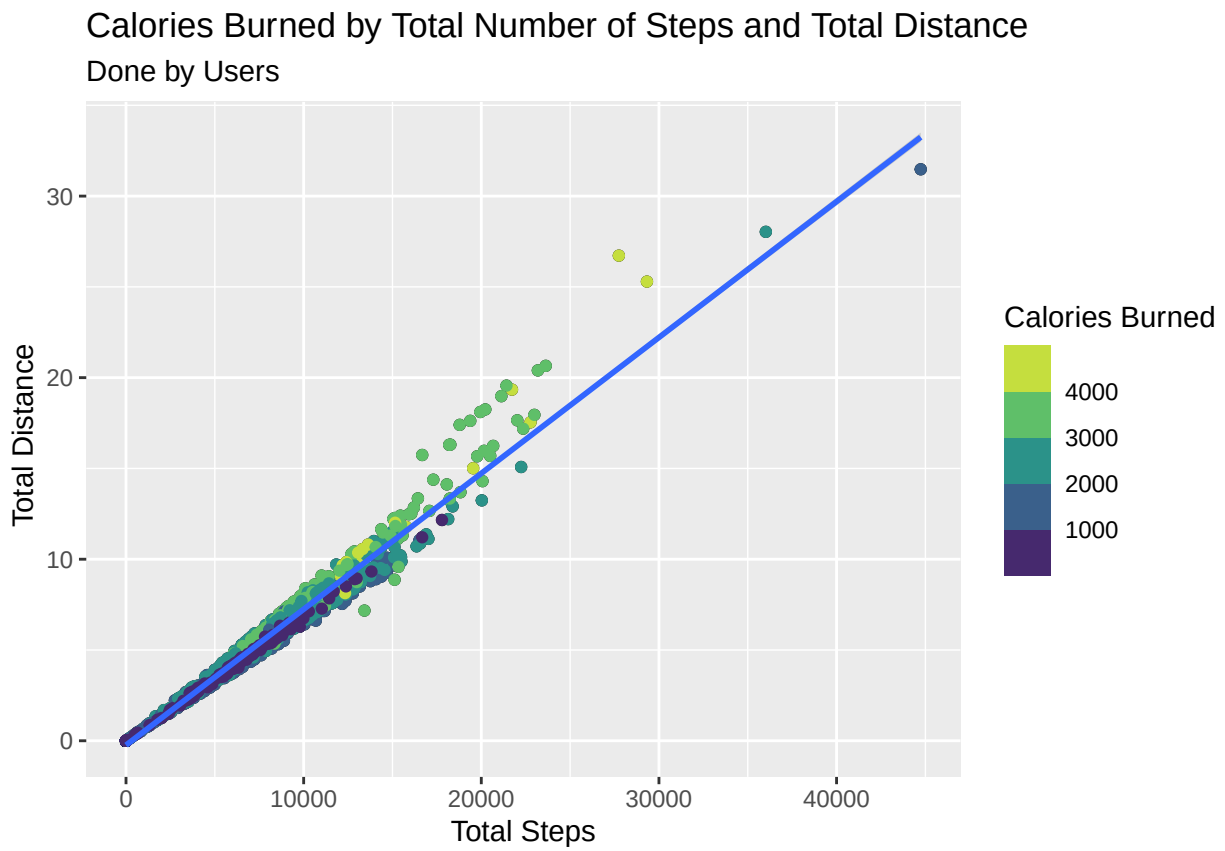
```
ggplot(activity, aes(total_steps, total_distance))+
  geom_jitter() +
  geom_point(aes(color = calories_burned)) +
  scale_color_viridis_b(name = "Calories Burned") +
  stat_smooth(method = lm) +
  labs(title = "Calories Burned by Total Number of Steps and Total Distance",
       subtitle = "Done by Users", x = "Total Steps", y = " Total Distance")
```

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

```
## Warning: Removed 96 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 96 rows containing missing values (geom_point).
```

```
## Warning: Removed 96 rows containing missing values (geom_point).
```




```
ggsave("calories_burned_by_total_steps.png")
```

```
## Saving 6.5 x 4.5 in image
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 96 rows containing non-finite values (stat_smooth).
```

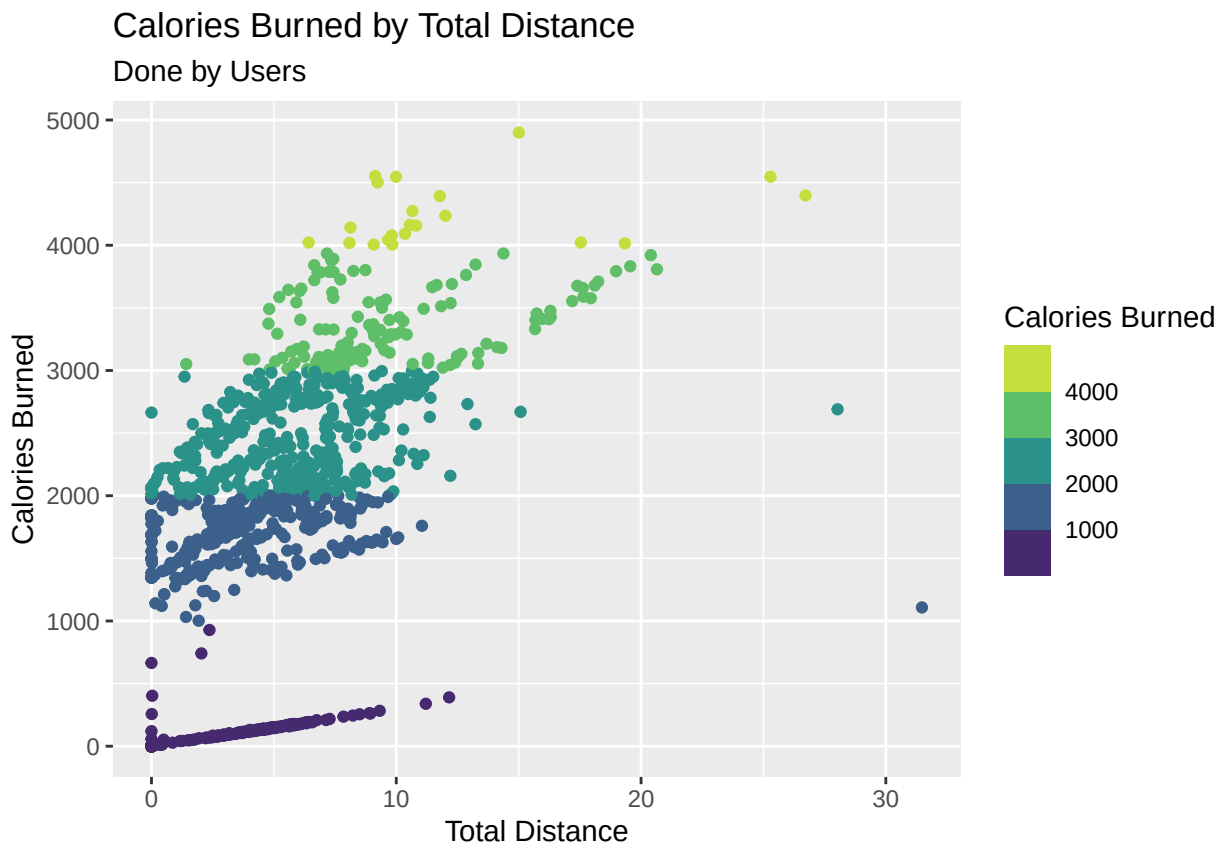
```
## Warning: Removed 96 rows containing missing values (geom_point).
```

```
## Warning: Removed 96 rows containing missing values (geom_point).
```

Finding: Strong corrilation of total steps taken with the total distance taken in terms of data relation. Along with calories burned the more steps and distance taken.

```
ggplot(data = activity, aes(x = total_distance, y = calories_burned)) +
  geom_point(aes(color = calories_burned)) +
  scale_color_viridis_b(name = "Calories Burned") +
  labs(title = "Calories Burned by Total Distance", subtitle = "Done by Users",
       x = "Total Distance", y = "Calories Burned")
```

```
## Warning: Removed 96 rows containing missing values (geom_point).
```




```
ggsave("calories_burned_by_total_distance.png")
```

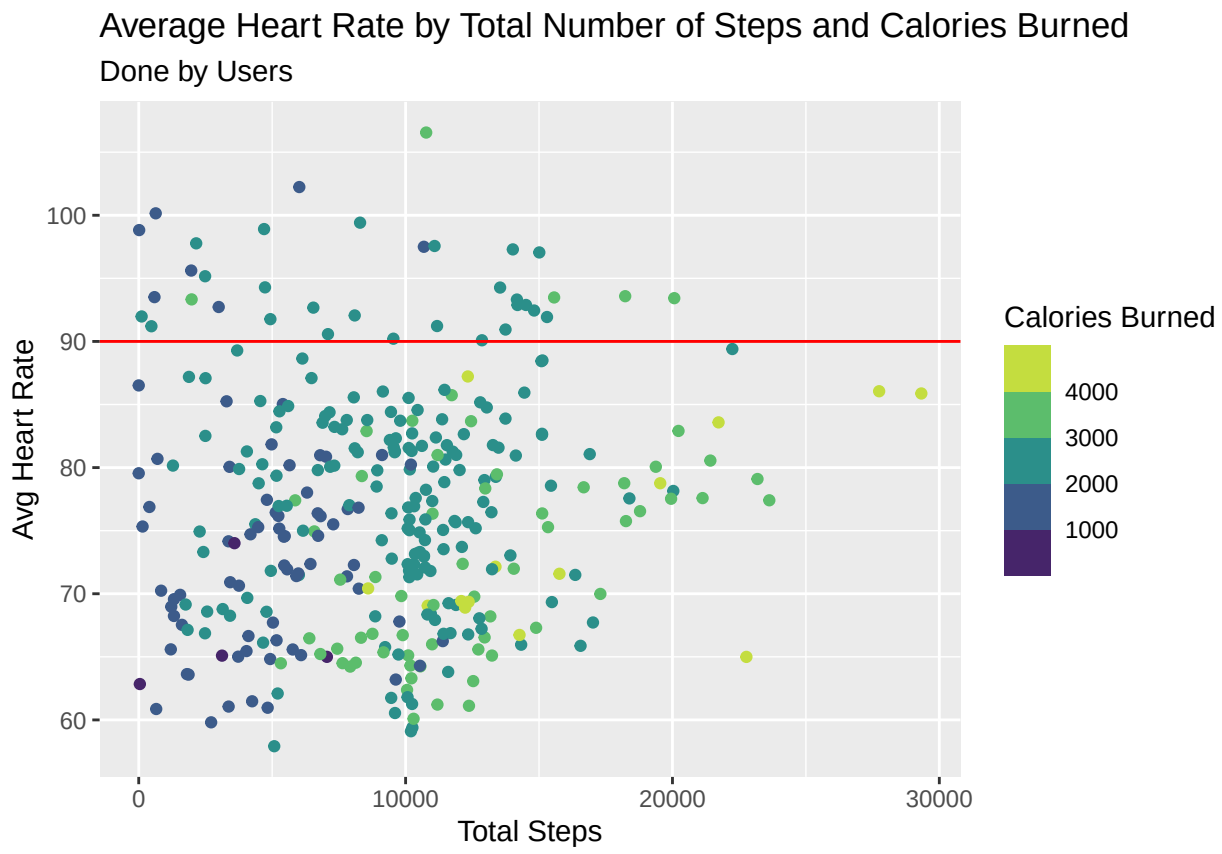
```
## Saving 6.5 x 4.5 in image
```

```
## Warning: Removed 96 rows containing missing values (geom_point).
```

Finding: Strong indication of two different user base. 1. This user base are power user who burn calories as intended by the amount of distance taken. 2. This user base take the most minimal distance to burn calories. However, these two segments is quite disproportionate.

```
heartrate %>%
  summarise(avg_heart_rate = mean(heart_rate)) %>%
  merge(activity, all = TRUE) %>%
  drop_na() %>%
  ggplot(aes(x = total_steps, y = avg_heart_rate)) +
  geom_point(aes(color = calories_burned)) +
  geom_hline(aes(yintercept = 90), color = "red") +
  scale_color_viridis_b(name = "Calories Burned") +
  labs(title = "Average Heart Rate by Total Number of Steps and Calories Burned",
       subtitle = "Done by Users", x = "Total Steps", y = "Avg Heart Rate")
```

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

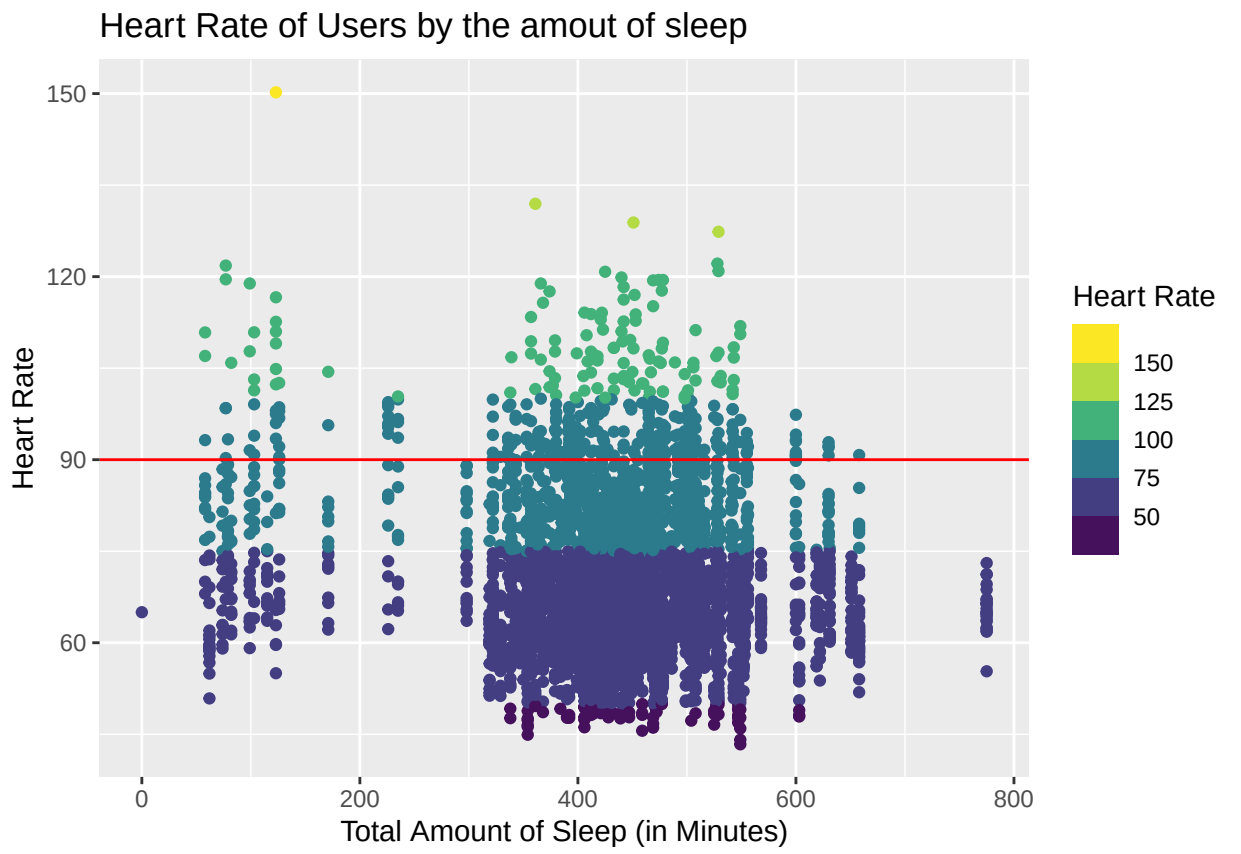



```
ggsave("avg_hear_rate_by_total_steps.png")
```

```
## Saving 6.5 x 4.5 in image
```

Findings: On average, most users' heart rate are not intensly high (below 90) to burn large amount of calories with a lot of amount of steps.

```
heartrate %>%
  #summarise(avg_heart_rate = mean(heart_rate)) %>%
  # The above line could be added to look at the average heart rate
  merge(sleep, all = TRUE) %>%
  drop_na() %>%
  ggplot(aes(x = total_minutes_asleep, y = heart_rate)) +
  geom_point(aes(color = heart_rate)) +
  geom_hline(aes(yintercept = 90), color = "red") +
  scale_color_viridis_b(name = "Heart Rate") +
  labs(title = "Heart Rate of Users by the amout of sleep",
       x = "Total Amount of Sleep (in Minutes)", y = "Heart Rate")
```




```
ggsave("heart_rate_by_sleep_amount.png")
```

```
## Saving 6.5 x 4.5 in image
```

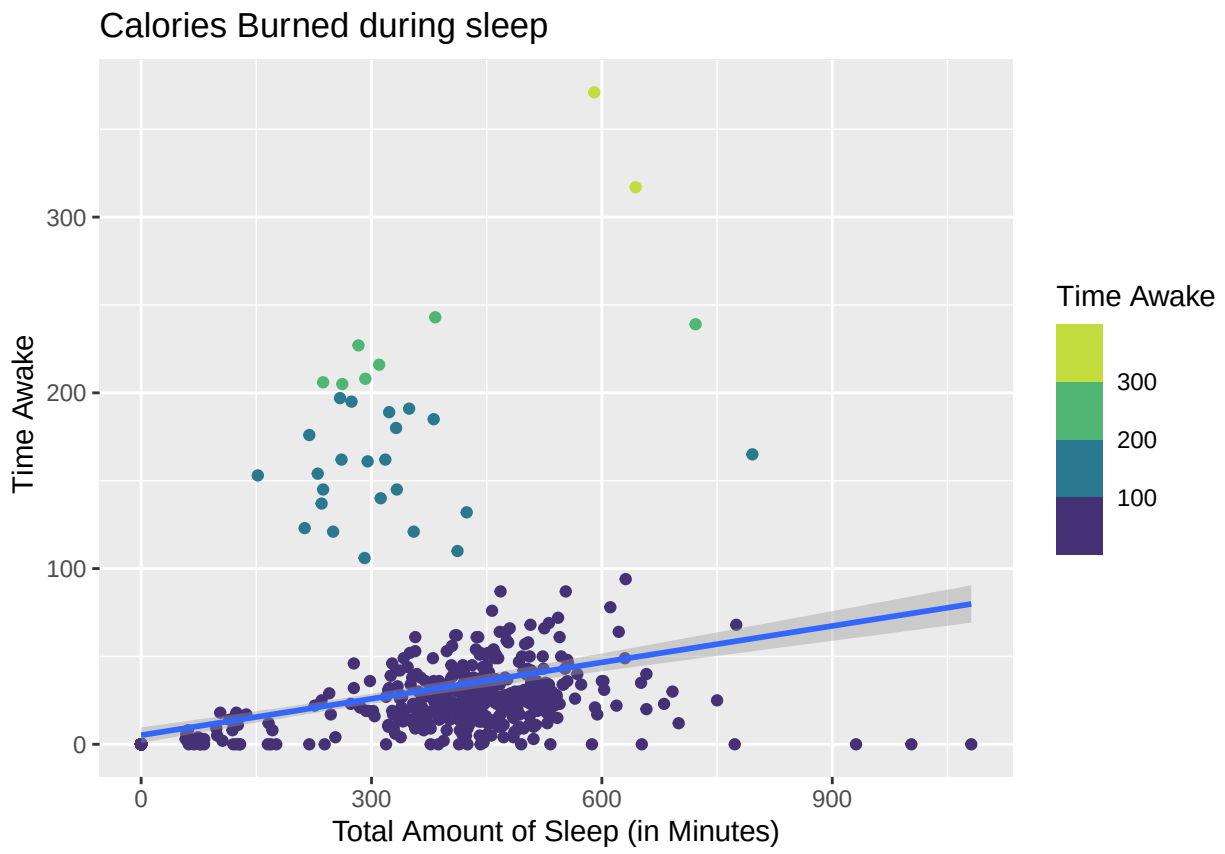
Finding: Majority of people who tracks their sleep data generally have low heart rate as expected, but have some outlier that have heart rate unusually high which could be false reading when a tracker is not equipped properly.

```
ggplot(data = sleep, aes(x = total_minutes_asleep, y = time_awake)) +
  geom_point(aes(color = time_awake)) +
  stat_smooth(method = lm, size = 1) +
  scale_color_viridis_b(name = "Time Awake") +
  labs(title = "Calories Burned during sleep",
       x = "Total Amount of Sleep (in Minutes)", y = " Time Awake")
```

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

```
## Warning: Removed 96 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 96 rows containing missing values (geom_point).
```




```
ggsave("calories_burned_during_sleep.png")
```

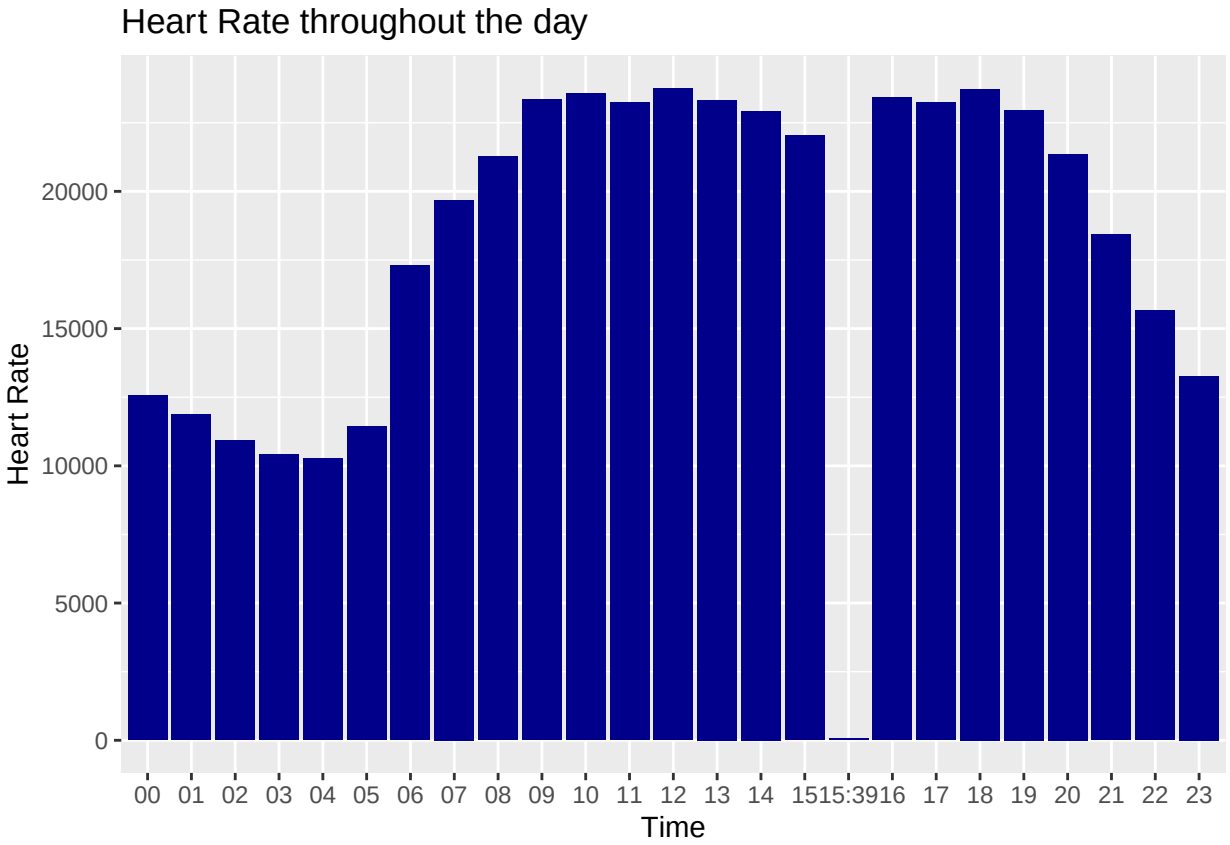
```
## Saving 6.5 x 4.5 in image
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 96 rows containing non-finite values (stat_smooth).

## Warning: Removed 96 rows containing missing values (geom_point).
```

Finding: Good indication that people generally stay asleep and do not wake up very much.

```
heartrate %>%
  merge(activity, all = TRUE) %>%
  drop_na() %>%
  ggplot(aes(x = time, y = heart_rate)) +
  geom_col(fill = "darkblue") +
  labs(title = "Heart Rate throughout the day",
       x = "Time", y = "Heart Rate")
```



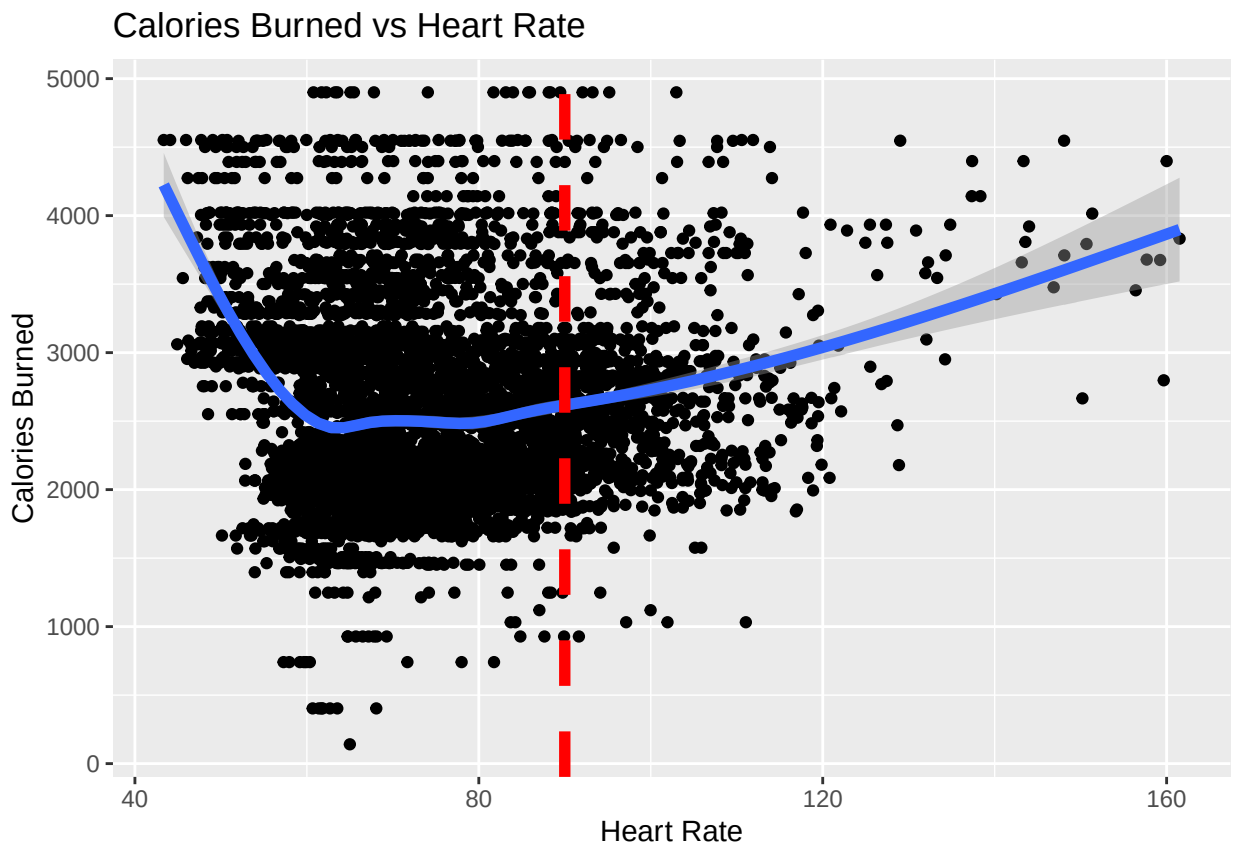

```
ggsave("heart_rate_in_a_day.png")
```

```
## Saving 6.5 x 4.5 in image
```

Finding: People are generally active throughout the day, but one discrepancy the graph inputted a time of 15:39.

```
heartrate %>%
  #summarise(avg_heart_rate = mean(heart_rate)) %>%
  merge(activity, all = TRUE) %>%
  drop_na() %>%
  ggplot(aes(x = heart_rate, y = calories_burned)) +
  geom_point() +
  geom_smooth(size = 2) +
  geom_vline(aes(xintercept = 90), color = "red", size = 2, linetype = "dashed") +
  labs(title = "Calories Burned vs Heart Rate",
       x = "Heart Rate", y = "Calories Burned") +
  theme(legend.position = "none")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
ggsave("calories_burned_vs_heart_rate.png")
```

```
## Saving 6.5 x 4.5 in image
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

Finding: There is indication that shows that more calories are burned with increase heart rate. However, this also shows that most users have a heart rate of less than 100 that burns between 2000 to 3000 calories. So it seems like users have moderate activities throught out the day as previously shown as well. ## Act

Recommendation

- Bellabeat can include functions in the Bellabeat app to alert users of their in-activity or unusual readings as timely notifications. Even a notification to indicate users to stretch their legs a little for in-activity to maintain wellness goals.
- Offer more or improved customization for users that regularly ask for the user's age, weight, height if the user chooses to input them to recommend users personalized tips to help the user achieve their wellness goals.
- Do a more targeted marketing campaign toward people who are more active, and health-conscious by showing the uniqueness of Bellabeat's products like the Spring (water bottle). For a broader marketing campaign, advertise the connection of each Bellabeat's products with getting enough activities every day, maintaining proper health and hydration, and wellness goals that other competitors can't.
- Points or rewards programs that both subscription and non-subscription-based users can earn for their activity to encourage users to continuously use Bellabeat products that they own while keeping users continue using the product to minimize forgetfulness which could potentially create brand loyalty.

Limitation

As previously mentioned:

- *Reliability:* **Low** - The datasets collected consisted of only 30 individuals who are anonymous with only the assumption that the majority of data collected are of the female gender.
- *Originality:* **Low** - The datasets were generated by respondents to a distributed survey via Amazon Mechanical Turk.
- *Comprehensive:* **High** - The datasets contain daily, hourly, and minutes of calories burned, activity intensity, number of steps, sleep duration, and weight information.
- *Current:* **Medium** - The datasets are 5 years old, but significant changes in a person's life may vary depending on a person's life events, habits, or routines. Data are recorded from 2016, March 12th to 2016, May 13th (3 months period).
- *Cited:* **Medium** - The data collection and source were well documented.
- The data collected do not indicate the user's age in order to indicate what is the appropriate heart rate for the individual as well.