BellaBeat - Google Data Analytics Capstone

Billy.Ljm

2023-02-08

Contents

0.1 Foreword

This is my capstone project for Google Data Analytics Professional Certificate.
I chose to do **Case Study 2: How Can a Wellness Technology Company Play it Smart?**.
I'll analyse the user data of *Bellabeat*: a hypothetical company designing fitness trackers for women.
The remainder of this report will be structured around the *six phases of Google's data analysis process*.

```
library(tidyverse)
library(lubridate)
library(waffle)

datadir = 'data/' # data directory
```

1 Ask

1.1 Background

Bellabeat's **current market niche** is:

- Focused on women from all countries
- Beautifully-designed smart products
- Tracks physical activity, mental state, menstrual cycle, etc
- Current products are a jewellery pendant *(Leaf)*, a watch *(Time)*, a water bottle *(Spring)*, and a companion app *(Bellabeat)*

The **current marketing strategy** to reach out to prospective users are:

- Sold solely through online store-fronts; no brick and mortar
- Extensive digital marketing *(Google Ads, YouTube, Facebook, Instagram, Twitter)*
- Some traditional advertising *(Radio, Billboards, Print, Television)*

1.2 Business Task

Bellabeat is going strong, but still would like to grow more via an updated marketing strategy.
The business task is to **identify themes/trends** in how people currently use their smart devices,
and translate them to high-level recommendations for **future marketing strategies**.

1.3 Stakeholders

The stakeholders of this analysis are:

- Bellabeat's Founders: *Urška Sršen and Sando Mur*
- Bellabeat Marketing Analytics Team

They should be kept in loop of the analysis's progress, and consulted if any unexpected situation arises.

2 Prepare

2.1 Dataset

The dataset used for this analysis is **FitBit Fitness Tracker Data** hosted on Kaggle or Zenodo.
It comprises `.csv` files of various fitness metrics measured from different users at different times, stored in a wide format.
The fitness tracker data was provided by 30 respondents to a paid distributed survey on Amazon Mechanical Turk in 2016.

2.2 Credibility

- **Not Reliable**: the small sample is unlikely to represent the whole population of Bellabeat's current and prospective users.

- **Not Original**: third party data source; from user to Amazon mechanical Turk to Zenodo to Kaggle
- **Not Current**: originated from 7 years ago, in a fast-moving market segment.
- **Not Cited**: data originated from a listing on Amazon mechanical Turk; there was no reputable entity behind it.
- **Comprehensive** (for its sample): measurements are automatically taken with a fitness tracker and seem complete at first glance.

Overall, I would have preferred to work on a more credible dataset such as Bellabeat's own user data.
This would have been more reliable, original, current, and cited; albeit data privacy would have to be carefully guarded.
However, since this is a hypothetical scenario and this is the only dataset available, I'll make do.

2.3 Data Privacy & Ethics

- **License**: Creative Commons Universal and Attribution, which have been complied with.
- **Privacy**: data is anonymised, and outdated by now.
- **Security**: already public available; does not require any security
- **Accessibility**: small sample would likely have excluded many groups *(but there are no alternate datasets)*

3 Process

3.1 Changelog

I'll combine the various spreadsheets into ~one dataset, so that the various fitness metric can be analysed in the context of each other.
However, different metrics are measured at different intervals; this is flagged by the second column being named `ActivityDate`, `ActivityHour`, `ActivitiyMinute`, and `Time` for measurement taken by day, hour, minute, and second respectively.
Thus, I'll have to aggregate seconds into minutes before combining with the other minute measurements, and so on. This will yield `data_sec`, `data_min`, `data_hr`, and `data_day`, which includes an increasing number of metrics but at coarser intervals.

A full list of changes made by the code below are:

- `minute***Wide_merged.csv` are ignored since they're reformatted duplicates of `minute***Narrow_merged.csv`. This files ignored are `minuteCaloriesWide_merged.csv`, `minuteIntensitiesWide_merged.csv`, `minuteStepsWide_merged.csv`
- `minuteSleep_merged.csv` is ignored, since the meaning of its `value` column is unknown; there are no accompanying dataset description
- `weightLogInfo_merged.csv` will be aggregated by the date (no time) of `Date`. Measurements on the same date will be averaged.
- `sleepDay_merged.csv` will be have the time section of `SleepDay` trimmed, leaving only the date.

- `data_sec` outer joins `heartrate_seconds_merged.csv` *(which is the only measurement taken every second)*

- `data_min` outer joins the aggregated `data_sec` with `minuteCaloriesNarrow_merged.csv`, `minuteIntensitiesNarrow_` `minuteMETsNarrow_merged.csv`, `minuteSleep_merged.csv`, and `minuteStepsNarrow_merged.csv`.

- `data_hr` outer joins the aggregated `data_min` with `hourlyCalories_merged.csv`, `hourlyIntensities_merged.csv`, and `hourlySteps_merged.csv`

- `data_day` outer joins the aggregated `data_hr` with `dailyActivity_merged.csv`, `dailyCalories_merged.csv`, `dailyIntensities_merged.csv`, `dailySteps_merged.csv`, `sleepDay_merged.csv`, and `weightLogInfo_merged.csv`.

- To aggregate, averaging will be done for `HeartRate`, `Intensity`, `METs`, `WeightKg`, `WeightPounds`, `Fat`, `BMI`; summing will be done for `Calories`, `Intensity`, `Steps`, `TotalDistance`, `LoggedActivitiesDistance`, `VeryActiveDistance`, `ModeratelyActiveDistance`, `LightActiveDistance`, `SedentaryActiveDistance`, `VeryActiveMinutes`, `FairlyActiveMinutes`, `LightlyActiveMinutes`, `SedentaryMinutes`, `TotalMinutesAsleep`, `TotalTimeInBed`

3.2 Seconds Data

The only metric measured every 5 seconds is heart rate,

```r
# read the only by second measurement
filename = 'heartrate_seconds_merged.csv'
data_sec = read_csv(paste(datadir, filename, sep=''), show_col_type=FALSE)
data_sec = rename(data_sec, DateTime=Time, HeartRate=Value)
data_sec$DateTime = as_datetime(data_sec$DateTime, format='%m/%d/%Y %I:%M:%S %p')
data_sec = group_by(data_sec, Id, DateTime)
glimpse(data_sec)
```

```
## Rows: 2,483,658
## Columns: 3
## Groups: Id, DateTime [2,483,658]
## $ Id        <dbl> 2022484408, 2022484408, 2022484408, 2022484408, 2022484408, ~
## $ DateTime  <dttm> 2016-04-12 07:21:00, 2016-04-12 07:21:05, 2016-04-12 07:21:~
## $ HeartRate <dbl> 97, 102, 105, 103, 101, 95, 91, 93, 94, 93, 92, 89, 83, 61, ~
```

3.3 Minutes Data

The metrics aggregated by minutes is,

```r
# aggregate data_sec
data_min = data_sec
data_min$DateTime = cut(data_min$DateTime, breaks='1 min')
data_min = summarise(data_min, HeartRate = mean(HeartRate, na.rm=TRUE),
                     .groups='keep')
data_min$DateTime = as_datetime(data_min$DateTime)

# outer join other datasets
for (filename in c('minuteCaloriesNarrow_merged.csv',
                  'minuteIntensitiesNarrow_merged.csv',
                  'minuteMETsNarrow_merged.csv',
                  'minuteStepsNarrow_merged.csv'))
{
  tmp = read_csv(paste(datadir, filename, sep=''), show_col_type=FALSE)
  colnames(tmp)[2] = 'DateTime'
  tmp$DateTime = as_datetime(tmp$DateTime, format='%m/%d/%Y %I:%M:%S %p')
  data_min = full_join(data_min, tmp, by=c('Id', 'DateTime'))
  rm(tmp)
}
rm(filename)

data_min = group_by(data_min, Id, DateTime)
glimpse(data_min)
```

```
## Rows: 1,325,854
## Columns: 7
## Groups: Id, DateTime [1,325,854]
## $ Id        <dbl> 2022484408, 2022484408, 2022484408, 2022484408, 2022484408, ~
## $ DateTime  <dttm> 2016-04-12 07:21:00, 2016-04-12 07:22:00, 2016-04-12 07:23:~
## $ HeartRate <dbl> 101.60000, 87.88889, 58.00000, 58.00000, 56.77778, 53.16667,~
## $ Calories  <dbl> 3.32064, 3.94326, 1.34901, 1.03770, 1.03770, 2.49048, 1.0377~
## $ Intensity <dbl> 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ METs      <dbl> 32, 38, 13, 10, 10, 24, 10, 26, 24, 26, 11, 34, 30, 34, 30, ~
## $ Steps     <dbl> 17, 9, 0, 0, 0, 7, 0, 13, 9, 15, 0, 43, 21, 30, 22, 42, 30, ~
```

3.4 Hourly Data

The metrics aggregated by hour are,

```
# aggregate data_min
data_hr = data_min
data_hr$DateTime = cut(data_hr$DateTime, breaks='1 h')
data_hr = summarise(data_hr, HeartRate = mean(HeartRate, na.rm=TRUE),
                    Calories = sum(Calories, na.rm=TRUE),
                    SumIntensity = sum(Intensity, na.rm=TRUE),
                    MeanIntensity = mean(Intensity, na.rm=TRUE),
                    METs = mean(METs, na.rm=TRUE),
                    Steps = sum(Steps, na.rm=TRUE), .groups='keep')
data_hr$DateTime = as_datetime(data_hr$DateTime)

# outer join other datasets
for (filename in c('hourlyCalories_merged.csv', 'hourlyIntensities_merged.csv',
                   'hourlySteps_merged.csv'))
{
  tmp = read_csv(paste(datadir, filename, sep=''), show_col_type=FALSE)
  colnames(tmp)[2] = 'DateTime'
  tmp$DateTime = as_datetime(tmp$DateTime, format='%m/%d/%Y %I:%M:%S %p')
  data_hr = full_join(data_hr, tmp, by=c('Id', 'DateTime'))
  rm(tmp)
}
rm(filename)

data_hr = group_by(data_hr, Id, DateTime)
glimpse(data_hr)
```

```
## Rows: 22,106
## Columns: 12
## Groups: Id, DateTime [22,106]
## $ Id               <dbl> 1503960366, 1503960366, 1503960366, 1503960366, 15039~
## $ DateTime         <dttm> 2016-04-12 00:00:00, 2016-04-12 01:00:00, 2016-04-12~
## $ HeartRate        <dbl> NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN~
## $ Calories.x       <dbl> 81.3241, 60.5605, 58.8302, 47.1900, 47.5046, 47.5046,~
## $ SumIntensity     <dbl> 20, 8, 7, 0, 0, 0, 0, 0, 13, 30, 29, 12, 11, 6, 36, 5~
## $ MeanIntensity    <dbl> 0.3333333, 0.1333333, 0.1166667, 0.0000000, 0.0000000~
## $ METs             <dbl> 17.23333, 12.83333, 12.46667, 10.00000, 10.06667, 10.~
## $ Steps            <dbl> 373, 160, 151, 0, 0, 0, 0, 0, 250, 1864, 676, 360, 25~
## $ Calories.y       <dbl> 81, 61, 59, 47, 48, 48, 48, 47, 68, 141, 99, 76, 73, ~
## $ 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~
## $ StepTotal        <dbl> 373, 160, 151, 0, 0, 0, 0, 0, 250, 1864, 676, 360, 25~
```

(note that the `HeartRate` row is `NA` since the upload session `Id` can change between different metrics)

It seems that the dataset had already aggregated `Calories`, `Intensity` and `Steps`.
We'll check if the different datasets agree to test the credibility of the data!

```
Calories = all(round(data_hr$Calories.x) == data_hr$Calories.y, na.rm=TRUE)
sumIntensity = all(data_hr$SumIntensity == data_hr$TotalIntensity, na.rm=TRUE)
MeanIntensity = all(round(data_hr$MeanIntensity, 5) ==
                      round(data_hr$AverageIntensity, 5), na.rm=TRUE)
Steps = all(data_hr$Steps == data_hr$StepTotal, na.rm=TRUE)

data_hr_test = data.frame(Calories, sumIntensity, MeanIntensity, Steps)
rm(Calories, sumIntensity, MeanIntensity, Steps)
glimpse(data_hr_test)
```

```
## Rows: 1
## Columns: 4
## $ Calories      <lgl> FALSE
## $ sumIntensity  <lgl> FALSE
## $ MeanIntensity <lgl> TRUE
## $ Steps         <lgl> FALSE
```

Great! No inconsistencies found across the dataset.
I'll just remove the duplicate columns and rename some of them.

```
data_hr = data_hr[,1:8]
colnames(data_hr) = gsub('.x', '', colnames(data_hr))
glimpse(data_hr)
```

```
## Rows: 22,106
## Columns: 8
## Groups: Id, DateTime [22,106]
## $ Id            <dbl> 1503960366, 1503960366, 1503960366, 1503960366, 15039603~
## $ DateTime      <dttm> 2016-04-12 00:00:00, 2016-04-12 01:00:00, 2016-04-12 02~
## $ HeartRate     <dbl> NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, N~
## $ Calories      <dbl> 81.3241, 60.5605, 58.8302, 47.1900, 47.5046, 47.5046, 47~
## $ SumIntensity  <dbl> 20, 8, 7, 0, 0, 0, 0, 0, 13, 30, 29, 12, 11, 6, 36, 58, ~
## $ MeanIntensity <dbl> 0.3333333, 0.1333333, 0.1166667, 0.0000000, 0.0000000, 0~
## $ METs          <dbl> 17.23333, 12.83333, 12.46667, 10.00000, 10.06667, 10.066~
## $ Steps         <dbl> 373, 160, 151, 0, 0, 0, 0, 0, 250, 1864, 676, 360, 253, ~
```

3.5 Daily Data

Lastly, the metrics aggregated by days are,

```
# aggregate data_hr
data_day = data_hr
data_day$DateTime = as_date(data_day$DateTime)
data_day = summarise(data_day, HeartRate = mean(HeartRate, na.rm=TRUE),
                     Calories = sum(Calories, na.rm=TRUE),
                     SumIntensity = sum(SumIntensity, na.rm=TRUE),
                     MeanIntensity = mean(MeanIntensity, na.rm=TRUE),
                     METs = mean(METs, na.rm=TRUE),
                     Steps = sum(Steps, na.rm=TRUE), .groups='keep')
data_day = rename(data_day, Date = DateTime)

# outer join other datasets
for (filename in c('dailyActivity_merged.csv', 'dailyCalories_merged.csv',
                   'dailyIntensities_merged.csv', 'dailySteps_merged.csv',
                   'sleepDay_merged.csv', 'weightLogInfo_merged.csv'))
{
  tmp = read_csv(paste(datadir, filename, sep=''), show_col_type=FALSE)
  colnames(tmp)[2] = 'Date'
  tmp$Date = as_date(tmp$Date, format='%m/%d/%Y')
  data_day = full_join(data_day, tmp, by=c('Id', 'Date'))
  rm(tmp)
}
rm(filename)

data_day = group_by(data_day, Id, Date)
colnames(data_day)
```

```
##  [1] "Id"                         "Date"
##  [3] "HeartRate"                  "Calories.x"
##  [5] "SumIntensity"               "MeanIntensity"
##  [7] "METs"                       "Steps"
##  [9] "TotalSteps"                 "TotalDistance"
## [11] "TrackerDistance"            "LoggedActivitiesDistance"
## [13] "VeryActiveDistance.x"       "ModeratelyActiveDistance.x"
## [15] "LightActiveDistance.x"      "SedentaryActiveDistance.x"
## [17] "VeryActiveMinutes.x"        "FairlyActiveMinutes.x"
## [19] "LightlyActiveMinutes.x"     "SedentaryMinutes.x"
## [21] "Calories.y"                 "Calories"
## [23] "SedentaryMinutes.y"         "LightlyActiveMinutes.y"
## [25] "FairlyActiveMinutes.y"      "VeryActiveMinutes.y"
## [27] "SedentaryActiveDistance.y"  "LightActiveDistance.y"
## [29] "ModeratelyActiveDistance.y" "VeryActiveDistance.y"
## [31] "StepTotal"                  "TotalSleepRecords"
## [33] "TotalMinutesAsleep"         "TotalTimeInBed"
## [35] "WeightKg"                   "WeightPounds"
## [37] "Fat"                        "BMI"
## [39] "IsManualReport"             "LogId"
```

Once again, it seems like there are columns with similar names and values.
However, this time it seems that these columns are not exactly equal. Let's quantify their difference

```
Calories1 = mean(data_day$Calories.x - data_day$Calories.y, na.rm=TRUE)
Calories2 = mean(data_day$Calories.x - data_day$Calories, na.rm=TRUE)
Steps = mean(data_day$Steps - data_day$TotalSteps, na.rm=TRUE)
Steps2 = mean(data_day$Steps - data_day$StepTotal, na.rm=TRUE)
```

```
Distance = mean(data_day$TotalDistance - data_day$TrackerDistance, na.rm=TRUE)
VeryActiveDistance = mean(data_day$VeryActiveDistance.x -
                            data_day$VeryActiveDistance.y, na.rm=TRUE)
ModeratelyActiveDistance = mean(data_day$ModeratelyActiveDistance.x -
                                  data_day$ModeratelyActiveDistance.y, na.rm=TRUE)
SedentaryActiveDistance = mean(data_day$SedentaryActiveDistance.x -
                                 data_day$SedentaryActiveDistance.y, na.rm=TRUE)
VeryActiveMinutes = mean(data_day$VeryActiveMinutes.x -
                           data_day$VeryActiveMinutes.y, na.rm=TRUE)
FairlyActiveMinutes = mean(data_day$FairlyActiveMinutes.x -
                             data_day$FairlyActiveMinutes.y, na.rm=TRUE)
LightlyActiveMinutes = mean(data_day$LightlyActiveMinutes.x -
                              data_day$LightlyActiveMinutes.y, na.rm=TRUE)
SedentaryMinutes = mean(data_day$SedentaryMinutes.x -
                          data_day$SedentaryMinutes.y, na.rm=TRUE)

data_day_test = data.frame(Calories1, Calories2, Steps, Steps2, Distance,
                           VeryActiveDistance, ModeratelyActiveDistance,
                           SedentaryActiveDistance, VeryActiveMinutes,
                           FairlyActiveMinutes, LightlyActiveMinutes,
                           SedentaryMinutes)
rm(Calories1, Calories2, Steps, Steps2, Distance, VeryActiveDistance,
   ModeratelyActiveDistance, SedentaryActiveDistance, VeryActiveMinutes,
   FairlyActiveMinutes, LightlyActiveMinutes, SedentaryMinutes)
glimpse(data_day_test)
```

```
## Rows: 1
## Columns: 12
## $ Calories1                <dbl> -14.66379
## $ Calories2                <dbl> -14.66379
## $ Steps                    <dbl> -112.7848
## $ Steps2                   <dbl> -112.7848
## $ Distance                 <dbl> 0.01435106
## $ VeryActiveDistance       <dbl> 0
## $ ModeratelyActiveDistance <dbl> 0
## $ SedentaryActiveDistance  <dbl> 0
## $ VeryActiveMinutes        <dbl> 0
## $ FairlyActiveMinutes      <dbl> 0
## $ LightlyActiveMinutes     <dbl> 0
## $ SedentaryMinutes         <dbl> 0
```

It seems there are some small differences between the aggregated `data_hr` values and the values directly pulled for the daily `.csv` files.
I'm not sure where this difference can come from. But no matter, the difference is small and we can should still be able to spot overall trends.

I'll just clean up the *approximately*-duplicate columns

```
data_day = data_day[,-(21:31)]
data_day = data_day[,-c(9, 11)]
colnames(data_day) = gsub('.x', '', colnames(data_day))
glimpse(data_day)
```

```
## Rows: 2,349
## Columns: 27
## Groups: Id, Date [966]
## $ Id                       <dbl> 1503960366, 1503960366, 1503960366, 150396036~
## $ Date                     <date> 2016-04-12, 2016-04-13, 2016-04-14, 2016-04-~
## $ HeartRate                <dbl> NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, ~
## $ Calories                 <dbl> 1985.205, 1797.860, 1776.625, 1745.243, 1863.~
## $ SumIntensity             <dbl> 429, 318, 293, 364, 349, 318, 391, 476, 313, ~
## $ MeanIntensity            <dbl> 0.2979167, 0.2208333, 0.2034722, 0.2527778, 0~
## $ METs                     <dbl> 17.52847, 15.87431, 15.68681, 15.40972, 16.45~
## $ Steps                    <dbl> 13158, 10735, 10460, 9685, 12669, 9705, 13019~
## $ TotalDistance            <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~
## $ TotalSleepRecords        <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ TotalMinutesAsleep       <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ TotalTimeInBed           <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ WeightKg                 <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ WeightPounds             <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ Fat                      <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ BMI                      <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ IsManualReport           <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ LogId                    <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
```

(note that there are rows full of `NA` since the upload session `Id` can change between different metrics)

4 Analyse

The business task is to identify trends to inform future **marketing strategies**.
Thus, I will want to focus on analysing how **different people** use their smart devices differently.
Since this might reveal what Bellabeat's current and prospective users want from their Bellabeat devices.
To do this, I would aggregate by upload session `Id`, which will be a proxy for different people *(assuming each metric is uploaded in a single session)*.

As for temporal trends, it seems that most would be irrelevant to our analysis.
For example, examining trends of calories burned over time might be useful for optimising a workout routine, which is relevant to the end user or as a new feature of the Bellabeat app, but is unlikely to be useful for crafting a marketing strategy.

4.1 Usage over Time

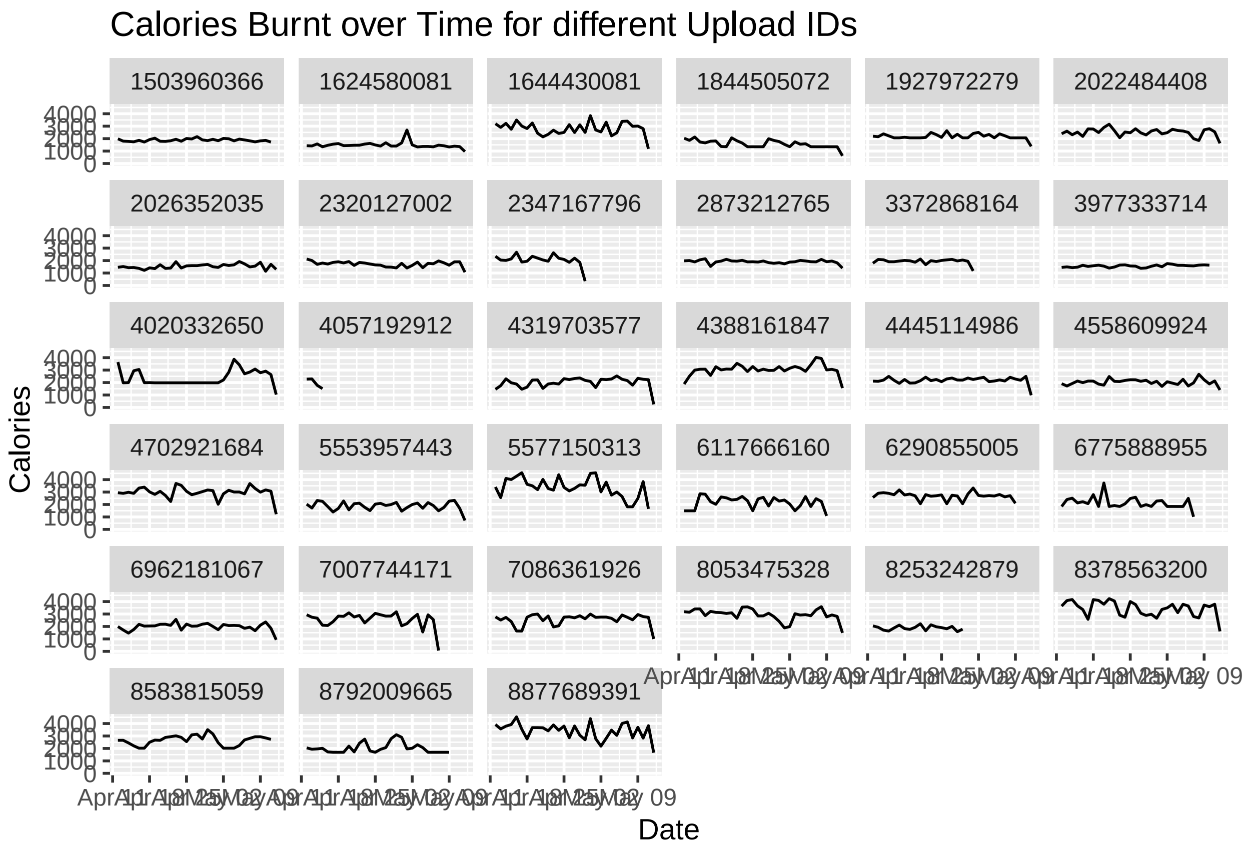
The one temporal trend that might be relevant is a change in usage of the smart device over time.
For example, a inhumanly low heart rate might suggest the user simply isn't wearing the device at that time.

```
sum(data_sec$HeartRate < 30)
```

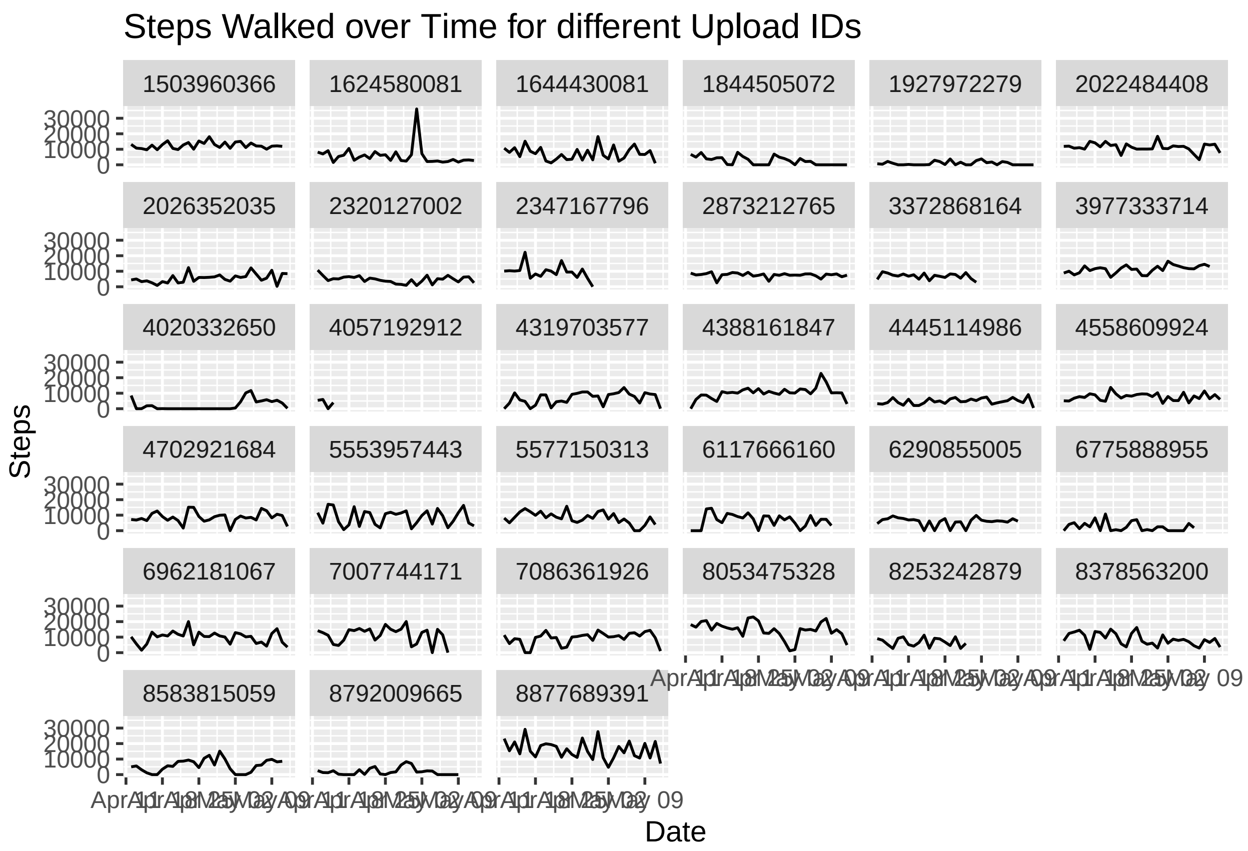
```
## [1] 0
```

Or a decrease in `Calories` or `Steps` over time might have resulted from users using their smart devices less.

```
ggplot(data_day, aes(x=Date, y=Calories)) + geom_line() + facet_wrap(~Id) +
  ggtitle("Calories Burnt over Time for different Upload IDs") +
  xlab("Date") + ylab("Calories")
```



```
ggplot(data_day, aes(x=Date, y=Steps)) + geom_line() + facet_wrap(~Id) +
  ggtitle("Steps Walked over Time for different Upload IDs") +
  xlab("Date") + ylab("Steps")
```



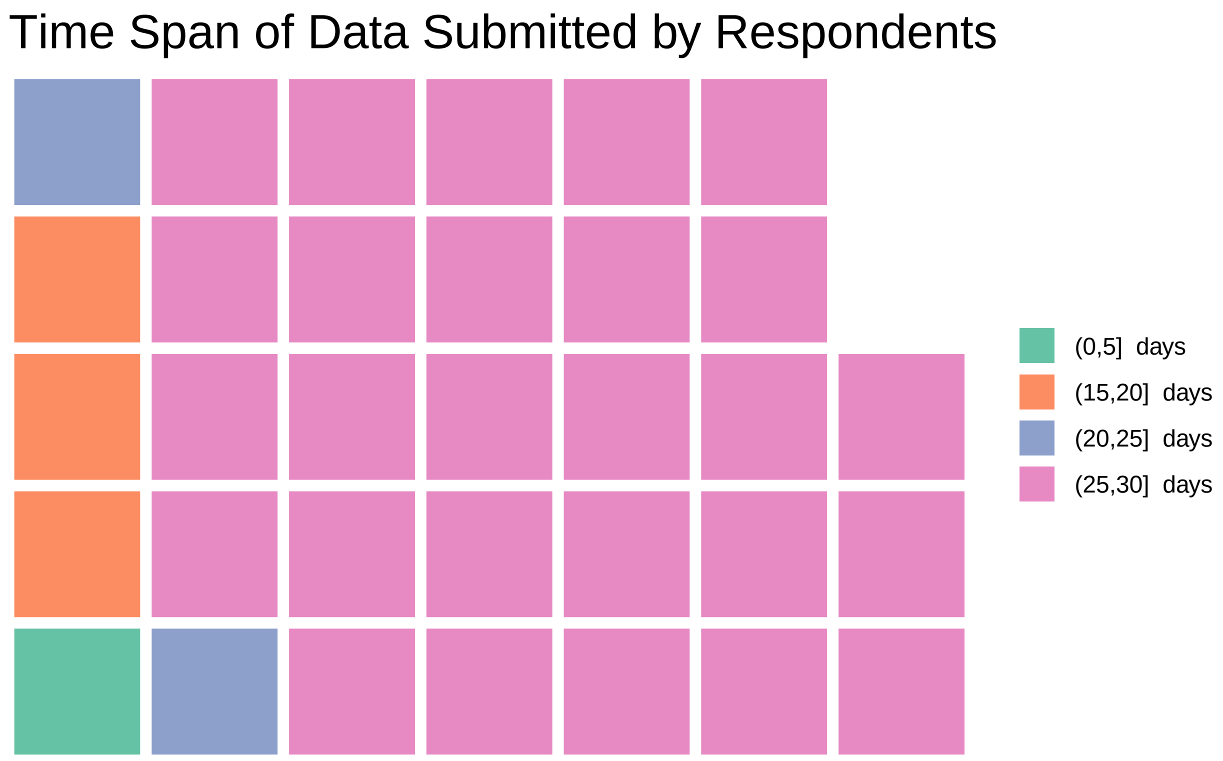
It seems that 12% (4 out of 33) of the upload sessions included data that stopped much earlier than the survey's date.
This might be because the users stopped using their smart devices, or that users simply didn't upload all their data.
The former suggests that **making a habit out of using smart devices might be difficult** for a significant proportion of people.
Thus, we might want to **highlight how effortless Bellabeat's devices are to use**, to appeal to this unfulfilled segment.

We can re-express this finding more intuitively in a different graph

```
time_span = group_by(data_day, Id)
time_span = summarise(time_span, UsageTime = max(Date, na.rm=TRUE) - min(Date, na.rm=TRUE))
time_span$UsageDays = cut(as.integer(time_span$UsageTime), breaks=c(0,5,10,15,20,25,30))
glimpse(time_span)
```

```
## Rows: 33
## Columns: 3
## $ Id        <dbl> 1503960366, 1624580081, 1644430081, 1844505072, 1927972279, ~
## $ UsageTime <drtn> 30 days, 30 days, 29 days, 30 days, 30 days, 30 days, 30 da~
## $ UsageDays <fct> "(25,30]", "(25,30]", "(25,30]", "(25,30]", "(25,30]", "(25,~
```

```
tmp = count(time_span, UsageDays)
time_span_plot = waffle(setNames(tmp$n, paste(tmp$UsageDays, " days")), rows=5,
                        title="Time Span of Data Submitted by Respondents")
rm(tmp)
time_span_plot
```



4.2 Variance b/w Users

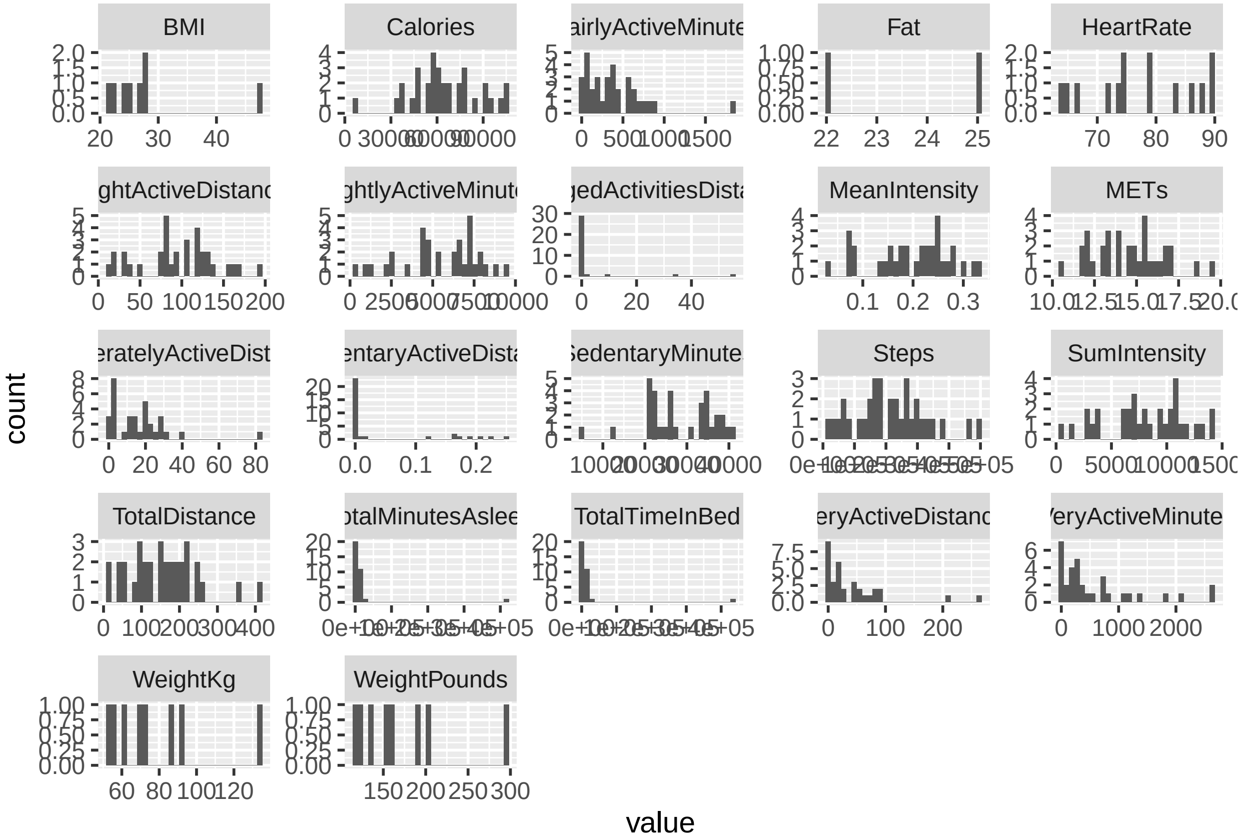
Since there are no other relevant temporal trends, we'll aggregate the data by time and focus on differences between users.
After aggregating by time, all the cleaned hourly/daily datasets would converge to the same time-aggregated data-set.

```
data_user = group_by(data_day, Id)
data_user = summarise(data_user, HeartRate = mean(HeartRate, na.rm=TRUE),
                    Calories = sum(Calories, na.rm=TRUE),
                    SumIntensity = sum(SumIntensity, na.rm=TRUE),
                    MeanIntensity = mean(MeanIntensity, na.rm=TRUE),
                    METs = mean(METs, na.rm=TRUE), Steps = sum(Steps, na.rm=TRUE),
                    TotalDistance = sum(TotalDistance, na.rm=TRUE),
                    LoggedActivitiesDistance = sum(LoggedActivitiesDistance, na.rm=TRUE),
                    VeryActiveDistance = sum(VeryActiveDistance, na.rm=TRUE),
                    ModeratelyActiveDistance = sum(ModeratelyActiveDistance, na.rm=TRUE),
                    LightActiveDistance = sum(LightActiveDistance, na.rm=TRUE),
                    SedentaryActiveDistance = sum(SedentaryActiveDistance, na.rm=TRUE),
                    VeryActiveMinutes = sum(VeryActiveMinutes, na.rm=TRUE),
                    FairlyActiveMinutes = sum(FairlyActiveMinutes, na.rm=TRUE),
                    LightlyActiveMinutes = sum(LightlyActiveMinutes, na.rm=TRUE),
                    SedentaryMinutes = sum(SedentaryMinutes, na.rm=TRUE),
                    TotalMinutesAsleep = sum(TotalMinutesAsleep, na.rm=TRUE),
                    TotalTimeInBed = sum(TotalTimeInBed, na.rm=TRUE),
                    WeightKg = mean(WeightKg, na.rm=TRUE),
                    WeightPounds = mean(WeightPounds, na.rm=TRUE),
                    Fat = mean(Fat, na.rm=TRUE), BMI = mean(BMI, na.rm=TRUE),
                    .groups='keep')
glimpse(data_user)
```

```
## Rows: 33
## Columns: 23
## Groups: Id [33]
## $ Id                       <dbl> 1503960366, 1624580081, 1644430081, 184450507~
## $ HeartRate                <dbl> NaN, NaN, NaN, NaN, NaN, 78.93425, 87.57059, ~
## $ Calories                 <dbl> 56289.869, 45951.972, 84218.786, 48746.976, 6~
## $ SumIntensity             <dbl> 11594, 5917, 7448, 3671, 1367, 12508, 7958, 6~
## $ MeanIntensity            <dbl> 0.26967923, 0.13365815, 0.17333333, 0.0822356~
## $ METs                     <dbl> 16.67202, 12.51649, 14.06676, 11.84516, 10.64~
## $ Steps                    <dbl> 374546, 177750, 217927, 79942, 28400, 351047,~
## $ TotalDistance            <dbl> 242.10, 121.36, 158.86, 52.89, 19.67, 250.61,~
## $ LoggedActivitiesDistance <dbl> 0.000000, 0.000000, 0.000000, 0.000000, 0.000~
## $ VeryActiveDistance       <dbl> 88.61, 29.12, 21.90, 0.26, 2.97, 75.07, 0.19,~
## $ ModeratelyActiveDistance <dbl> 24.62, 11.18, 28.53, 1.52, 0.97, 22.32, 0.35,~
## $ LightActiveDistance      <dbl> 128.74, 80.81, 108.27, 51.07, 15.72, 153.22, ~
## $ SedentaryActiveDistance  <dbl> 0.00, 0.19, 0.12, 0.00, 0.00, 0.00, 0.00, 0.0~
## $ VeryActiveMinutes        <dbl> 1200, 269, 287, 4, 41, 1125, 3, 42, 243, 437,~
## $ FairlyActiveMinutes      <dbl> 594, 180, 641, 40, 24, 600, 8, 80, 370, 190, ~
## $ LightlyActiveMinutes     <dbl> 6818, 4758, 5354, 3579, 1196, 7981, 7956, 614~
## $ SedentaryMinutes         <dbl> 26293, 38990, 34856, 37405, 40840, 34490, 213~
## $ TotalMinutesAsleep       <dbl> 18014, 0, 1176, 1956, 2085, 0, 14173, 61, 670~
## $ TotalTimeInBed           <dbl> 19160, 0, 1384, 2883, 2189, 0, 15054, 69, 737~
## $ WeightKg                 <dbl> 52.60000, NaN, NaN, NaN, 133.50000, NaN, NaN,~
## $ WeightPounds             <dbl> 115.9631, NaN, NaN, NaN, 294.3171, NaN, NaN, ~
## $ Fat                      <dbl> 22, NaN, NaN, NaN, NaN, NaN, NaN, NaN, NaN, N~
## $ BMI                      <dbl> 22.650, NaN, NaN, NaN, 47.540, NaN, NaN, NaN,~
```

Lets look at the distribution of the fitness metrics across different users.

```
ggplot(pivot_longer(data_user, colnames(data_user)[-1]), aes(x = value)) +
       geom_histogram() + facet_wrap(.~name, scales='free')
```



The distributions generally look normal with just one peak.
Thus, the dataset suggests that there **isn't much segmentation of the users of fitness trackers**.
And future marketing strategies can **target prospective users as one cohesive group**.

4.3 User Behavior

Let's try to see how this one cohesive group generally behaves.
I'll average the metrics across all the users, since their distributions has been visually verified to be normal.

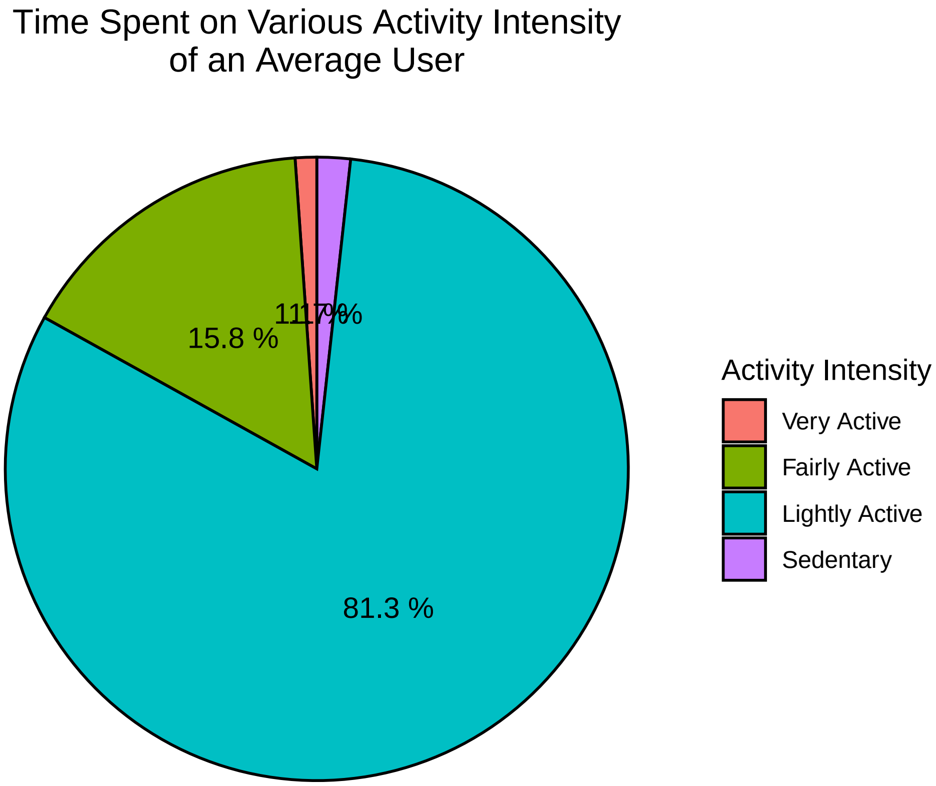
```
data_avg = data_user[-1]
data_avg = summarise(data_avg, across(everything(),
                                      list(function(x) mean(x, na.rm=TRUE))
                                    ))
colnames(data_avg) = gsub('_1', '', colnames(data_avg))
glimpse(data_avg)
```

```
## Rows: 1
## Columns: 22
## $ HeartRate                <dbl> 77.45672
## $ Calories                 <dbl> 65199.64
## $ SumIntensity             <dbl> 8057.667
## $ MeanIntensity            <dbl> 0.1963962
## $ METs                     <dbl> 14.57742
## $ Steps                    <dbl> 214350
## $ TotalDistance            <dbl> 156.3733
## $ LoggedActivitiesDistance <dbl> 3.081233
## $ VeryActiveDistance       <dbl> 42.80364
## $ ModeratelyActiveDistance <dbl> 16.16636
## $ LightActiveDistance      <dbl> 95.16273
## $ SedentaryActiveDistance  <dbl> 0.04575758
## $ VeryActiveMinutes        <dbl> 602.8788
## $ FairlyActiveMinutes      <dbl> 386.3939
## $ LightlyActiveMinutes     <dbl> 5492.242
## $ SedentaryMinutes         <dbl> 28234.48
## $ TotalMinutesAsleep       <dbl> 18180.12
## $ TotalTimeInBed           <dbl> 19209.06
## $ WeightKg                 <dbl> 77.81115
## $ WeightPounds             <dbl> 171.5442
## $ Fat                      <dbl> 23.5
## $ BMI                      <dbl> 27.98801
```

4.3.1 Activity Intensity

Firstly, the `MeanIntensity` is pretty low at 0.2, which suggests that the fitness tracker users do not usually do consistent high-intensity sports.
This can be corroborated by comparing their Very, Fairly, Lightly and Sedentary minutes below.

```
intensity_plot =
  ggplot(gather(data_avg[c('VeryActiveMinutes', 'FairlyActiveMinutes',
                         'LightlyActiveMinutes', 'SedentaryMinutes')]),
                      aes(x='', y=value, fill=key)) +
  geom_bar(stat="identity", width=1, color="black") +
  coord_polar("y", start=0) +
  geom_text(aes(label = paste(round(value/sum(value) * 100, 1), '%')),
          position = position_stack(vjust = 0.5)) +
  guides(fill = guide_legend(title = "Activity Intensity")) +
  scale_fill_discrete(labels = c("Very Active", "Fairly Active",
                                 "Lightly Active", "Sedentary")) +
  ggtitle("Time Spent on Various Activity Intensity\nof an Average User") +
  theme_void() + theme(plot.title = element_text(hjust = 0.5))

intensity_plot
```

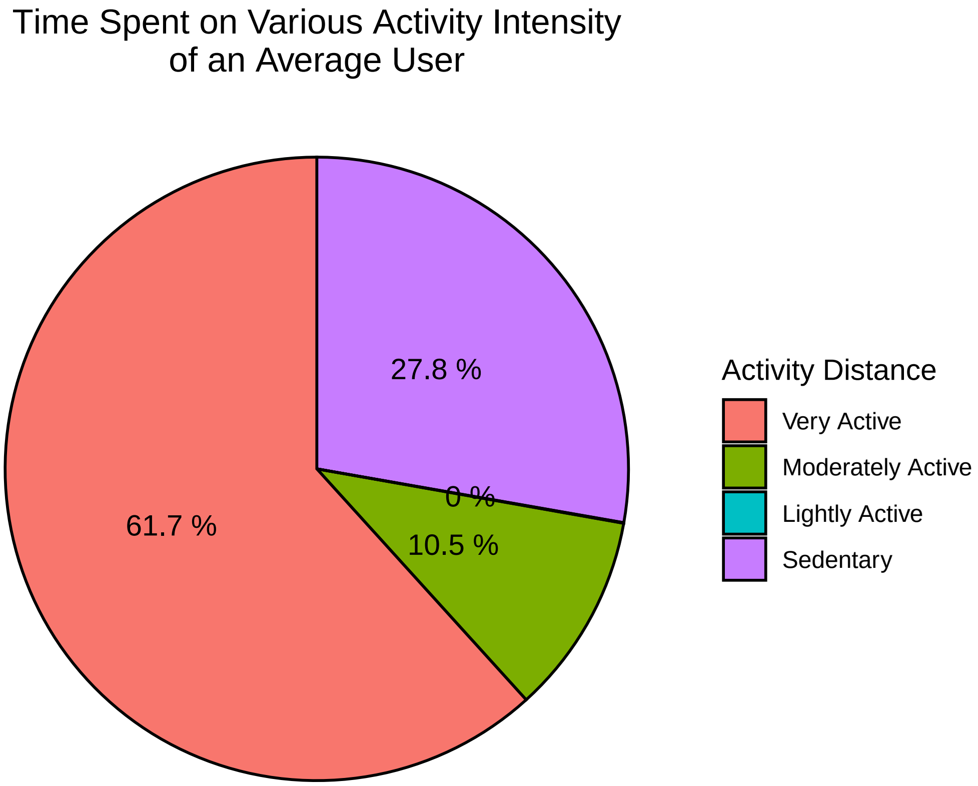


It seems that the average user **spends an large majority of their time on lightly-intensity activities**. Thus, marketing materials should **highlight features/metrics that are relevant** to light-intensity activities.

4.3.2 Activity Type

Some additional insight can be gained from comparing the distance moved during the above-mentioned activities.
From the scale of the histograms plotted above, we notice that a much larger distance is travelled during "High Intensity Activities".
We can see this most strikingly when plotting the percentage of distance travelled.

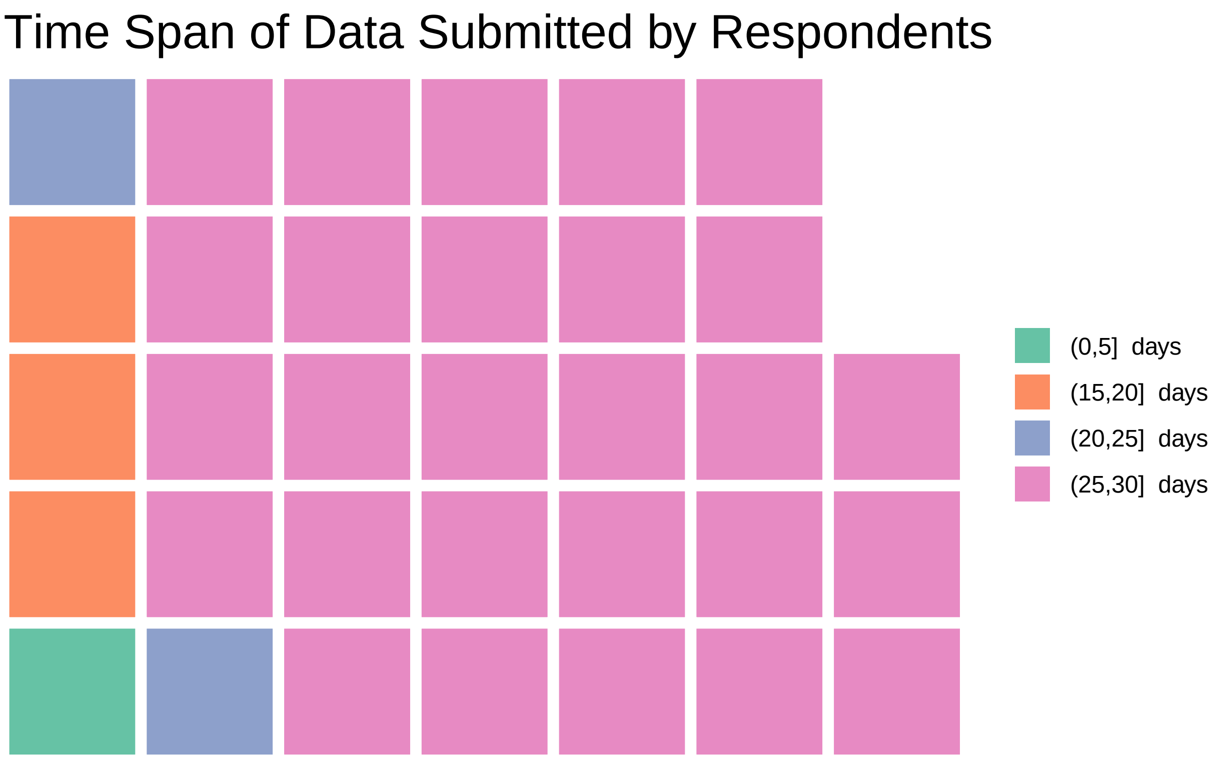
```
distance_plot =
  ggplot(gather(data_avg[c('VeryActiveDistance', 'ModeratelyActiveDistance',
                         'LightActiveDistance', 'SedentaryActiveDistance')]),
                      aes(x='', y=value, fill=key)) +
  geom_bar(stat="identity", width=1, color="black") +
  coord_polar("y", start=0) +
  geom_text(aes(label = paste(round(value/sum(value) * 100, 1), '%')),
            position = position_stack(vjust = 0.5)) +
  guides(fill = guide_legend(title = "Activity Distance")) +
  scale_fill_discrete(labels = c("Very Active", "Moderately Active",
                                 "Lightly Active", "Sedentary")) +
  ggtitle("Time Spent on Various Activity Intensity\nof an Average User") +
  theme_void() + theme(plot.title = element_text(hjust = 0.5))

distance_plot
```



This seems to suggest that **users overwhelming prefer non-static workouts**.
Thus, Bellabeta can **highlight features for jogging and other non-static workouts**, over other static workouts like gymming.

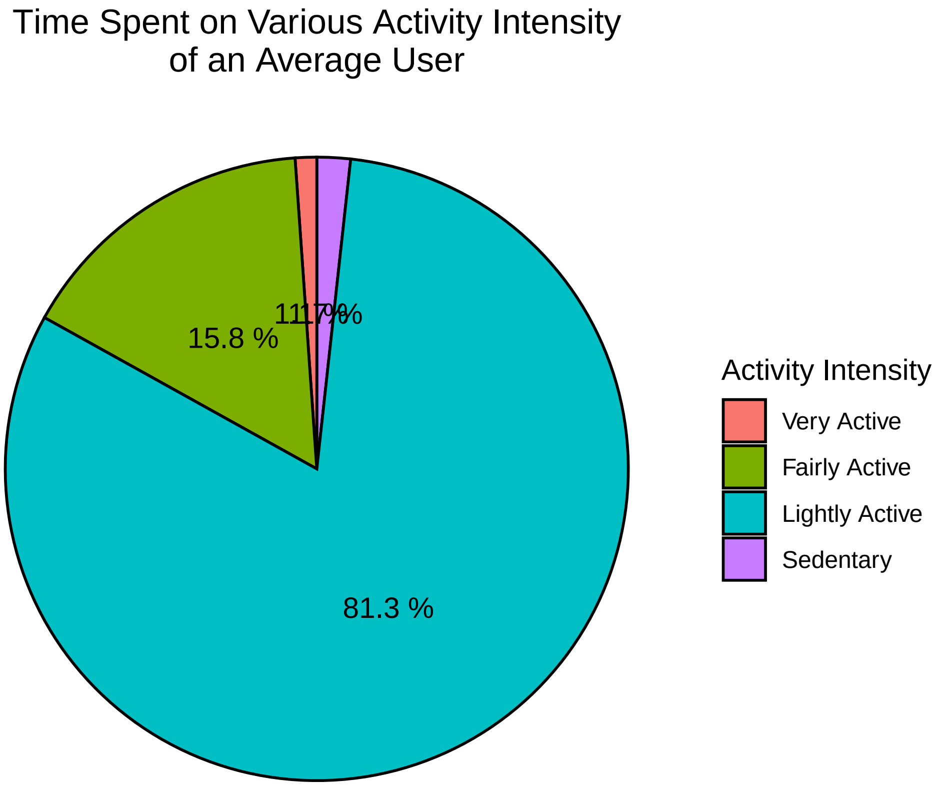
5 Share

We noticed that 12% (4 out of 33) of the survey respondents did not use their fitness tracker over the usual 30 days.
Thus, it seems that their fitness tracking needs are still unmet by our products, and they still represent a untapped market segment.

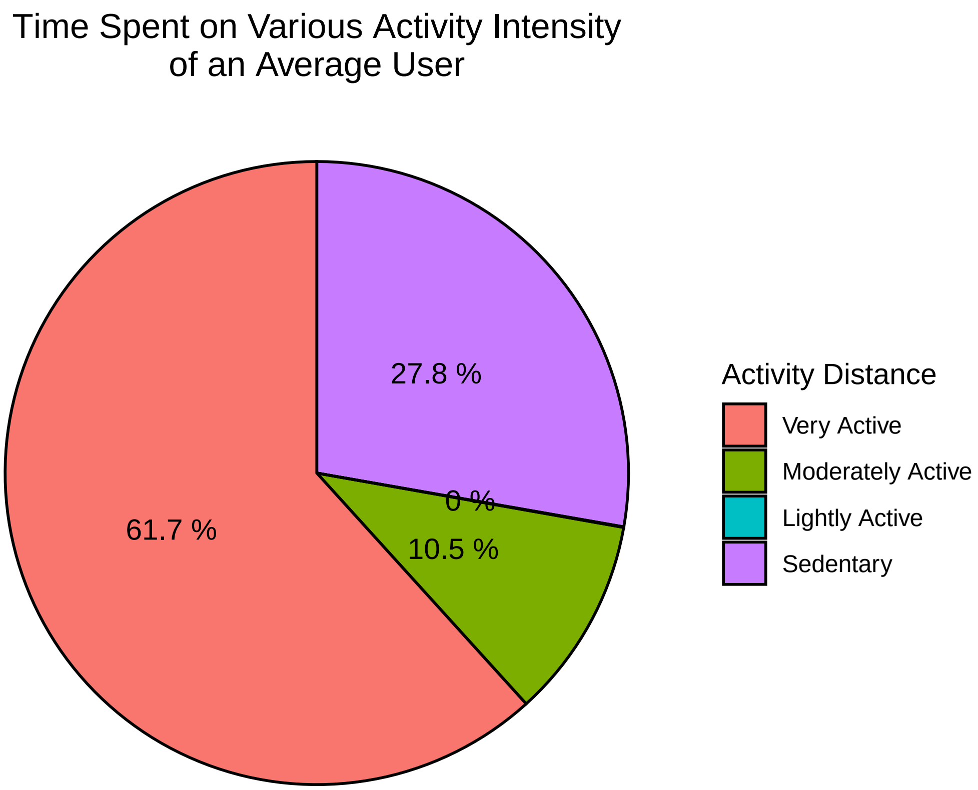


Additionally, we not exhibit any segmentation in the fitness metrics of the survey respondents, suggesting that Bellabeat’s users tend to interact with their smart device in similar ways *(without segmentation into athletes, etc)*.

For this prototypical Bellabeat user, we discovered that they spent most of their time engaged in light-intensity activities.



However, when they do engage in higher-intensity activities, they cover a lot of ground, suggesting that these higher-intensity activities are non-static.



6 Act

The key takeaways of this analysis are:

1. There is a significant segment of users who bought Bellabeat's product, but have not been regularly using them.
   Thus, we could consider marketing to them via more cost-effective means such as push notifications in the Bellabeat app.

2. Bellabeat's users spend an overwhelming majority of their time in light-intensity activities.
   Thus, our marketing should highlight features for light-intensity activities such as stress tracking.

3. However, when Bellabeat users do workout, they tend to do non-static exercises.
   Thus, our marketing should focus on features for jogging and other non-static workouts, over static workouts like gymming.

As a further direction of analysis, we could also consider how to market Bellabeat's new products to existing users.
For example, how should we convince *Bellabeat leaf* users to purchase the *Bellabeat time*.
This would require data that is segmented according to the different products.