Bellabeat Case Study

Alexander Gandji

2022-06-06

Contents

Introduction

Bellabeat, a high-tech company that manufactures health-focused smart products wants to analyse the usage of one of their products in order to gain insight into how people are already using their smart devices.Then,using this information, she would like high-level recommendations for how these trends can inform Bellabeat marketing strategy. The main focus of this case is to analyze smart devices fitness data and determine how it could help unlock new growth opportunities for Bellabeat.

Key Stakeholders:

- Primary Stakeholders:
- UrškaSršen: Bellabeat's cofounder and Chief Creative Officer.
- SandoMur: Mathematician and Bellabeat's cofounder; key member of the Bellabeat executive team.
- Secondary Stakeholders:
- Bellabeat Marketing Analytics Team: A team of data analysts responsible for collecting, analyzing, and reporting data that helps guide Bellabeat's marketing strategy.

Ask phase

Business Task

Analyzing data of smart devices to identify current trends, which can be applied to users, to drive Bellabeat's/our new marketing strategy

Who are the stake holders

- Urška Sršen=> Bellabeat's cofounder and Chief Creative Officer
- Sando Mur=> Mathematician and Bellabeat's cofounder; key member of the Bellabeat executive team
- Bellabeat marketing analytics team=> A team of data analysts responsible for collecting, analyzing, and reporting data that helps guide Bellabeat's marketing strategy

Prepare Phase

Where is the data stored?

The data is stored on Kaggle and provided by Mönius. The data was gathered through a survey via Amazon Mechanical Tusk. The survey has 33 eligible participants.

Data limitation

We are dealing with a small sample size of only 33 participants, which is not a representative sample size. Furthermore, the data set does not provide any information regarding the following aspects;

- Social background of the participants (education, profession etc.)
- When the participants were active, what activity were they doing?
- Where was the study performed?

- There is no information about the participant's gender The weight loss data set was not included because only data from 8 participants is available

Reliability of the data

Reliability & originality The data set is provided by the user Mönius. The user-provided a link to the original source of the data=> https://zenodo.org/record/53894#.X9oeh3Uzaao

Comprehensivness This aspect is given because the location of the data set and the link to the data source provide enough information;

- Name of the company
- How many participants participated
- Definition of each variable

Current The data was collected in 2016(in the case study was no data mentioned therefore the assumption was made that the case study is from 2016/2017, which makes the data current)

Cited The link to the original source is provided by the Kaggle user Mönius. The authors are mentioned on the website where the original data set is located, the authors are mentioned. On the Kaggle website, it is said who collected the data and in what context were the data collected.

Process Phase

The cleaning process was performed with two tools, Excel Spreadsheet and R

Excel Spreadsheet

The spreadsheet was selected because the data sets did not contain an overwhelming amount of data; therefore, the data was sorted and filtered in a spreadsheet to do the initial data screening. In this process following data values were deleted;

- Id=> 1503960366, ActivityDate=> 12.05.2016
- Id=> 6290855005, ActivityDate=> 10.05.2016
- Id=> 8253242879, ActivityDate=> 30.04.2016
- Id=> 8583815059, ActivityDate=> 12.05.2016 These values were removed from the dailyActivity data set because they did not gather any data besides 1440 minutes of sedentary data, which could mean the participants forgot to wear their smart devices that day.

R

First, all essential packages for the cleaning and analysis process were installed and added to the library.

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```r
install.packages("dplyr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```r
install.packages("ggplot2")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```r
install.packages("formatR")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(skimr)
library(dplyr)
library(ggplot2)
library(formatR)
```

Upload of files

Overview of distinct data values

```r
nrow(daily_activity)
```

```
## [1] 936
```

```r
nrow(daily_intensities)
```

```
## [1] 940
```

```r
nrow(daily_sleep)
```

```
## [1] 413
```

```r
nrow(hourly_calories)
```

```
## [1] 22099
```

```r
nrow(hourly_intensities)
```

```
## [1] 22099
```

```r
nrow(hourly_steps)
```

```
## [1] 22099
```

```
nrow(minute_sleep)
```

```
## [1] 188521
```

to get an overall idea, it is also important to check if all sets (at least the sets with the same time range like daily, hourly etc.)

```
n_distinct(daily_activity$Id)
```

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

```
n_distinct(daily_intensities$Id)
```

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

```
n_distinct(daily_sleep$Id)
```

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

```
n_distinct(hourly_calories$Id)
```

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

```
n_distinct(hourly_intensities$Id)
```

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

```
n_distinct(hourly_steps$Id)
```

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

```
n_distinct(minute_sleep$Id)
```

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

Cleaning the data by dropping all NAs

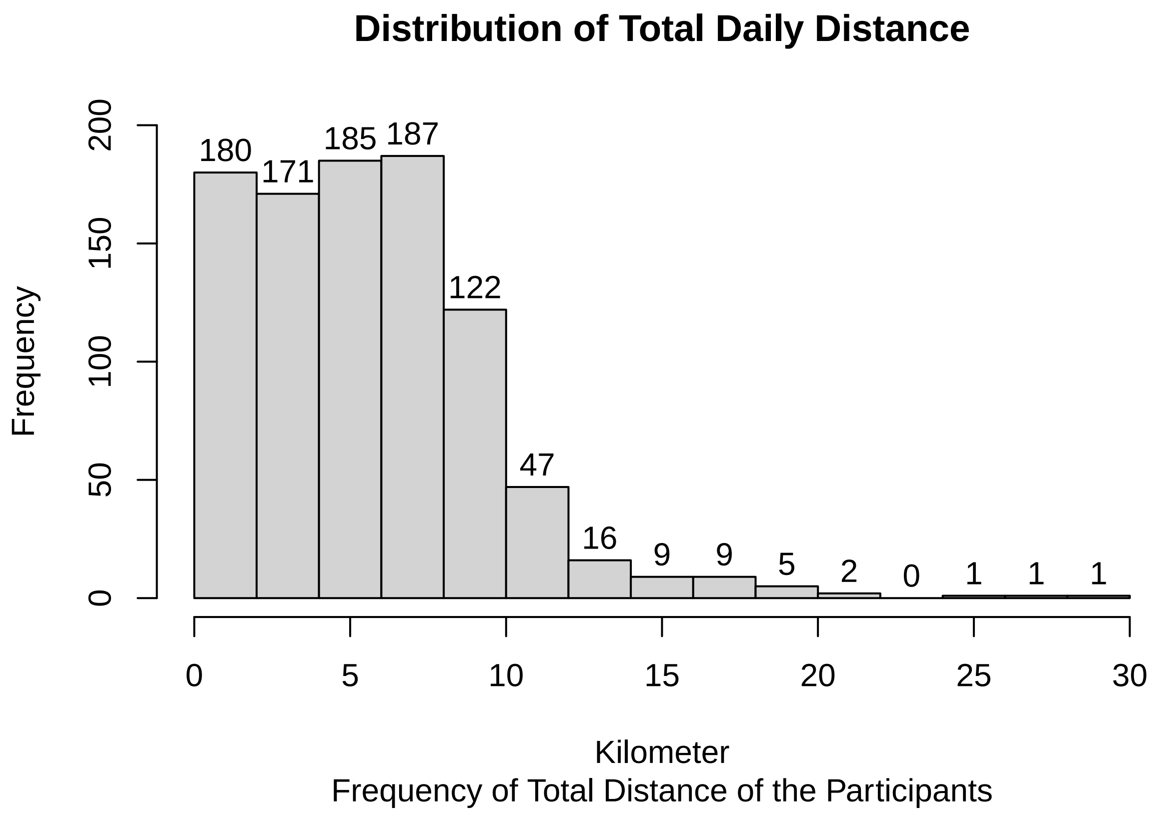
Even after the prescreening in the Excel Spreadsheet, the whole process was done again in R by dropping NA values.

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

daily_intensities_clean <- daily_intensities %>%
  drop_na()

hourly_calories_clean <- hourly_calories %>%
  drop_na()

hourly_intensities_clean <- hourly_intensities %>%
  drop_na()

hourly_steps_clean <- hourly_steps %>%
  drop_na()

minute_sleep_clean <- minute_sleep %>%
  drop_na()

daily_sleep_clean <- daily_sleep %>%
  drop_na()
```

Analysis & Share Process

The first part of the analysis will give an outline of the participants behavior with descriptive statistics

```
total_distance_da <- daily_activity_clean %>%
    select(TotalDistance)

total_distance_da_ <- as.numeric(unlist(total_distance_da))

hist(total_distance_da_, xlab = "Kilometer", labels = TRUE, ylim = c(0,
    200), main = "Distribution of Total Daily Distance", sub = "Frequency of Total Distance of the Parti
```



The histogram shows us that the majority of the data occurs between 0-10 Kilometers to see if we are right we test that statistically

```
mode <- function(x) {
    u <- unique(x)
    tab <- tabulate(match(x, u))
    u[tab == max(tab)]
}

total_distance_da %>%
    summarize(mean = max(total_distance_da_), median = median(total_distance_da_),
        mode = mode(total_distance_da_), min = min(total_distance_da_),
        max = max(total_distance_da_))
```

```
## # A tibble: 1 x 5
##    mean median  mode   min   max
##   <dbl>  <dbl> <dbl> <dbl> <dbl>
## 1  28.0   5.27     0     0  28.0
```

The calculations show that it is a right-skewed distribution.
What does that mean for Bellabeat? It gives an indicator of what number of kilometres has been reached the most, which will help for the clustering.

Cluster

The histogram gave an indicator of how the data is distributed. Before a cluster is created, it is essential to analyse the already existing level of how the kilometres have been accumulated.

```
## Sedentary
daily_activity_clean %>%
    summarize(mean = mean(daily_activity_clean$SedentaryActiveDistance),
        min = min(daily_activity_clean$SedentaryActiveDistance),
        max = max(daily_activity_clean$SedentaryActiveDistance))
```

```
## # A tibble: 1 x 3
##      mean   min   max
##     <dbl> <dbl> <dbl>
## 1 0.00161     0 0.110
```

```
## LightActivity
daily_activity_clean %>%
    summarize(mean = mean(daily_activity_clean$LightActiveDistance),
        min = min(daily_activity_clean$LightActiveDistance),
        max = max(daily_activity_clean$LightActiveDistance))
```

```
## # A tibble: 1 x 3
##    mean   min   max
##   <dbl> <dbl> <dbl>
## 1  3.36     0  10.7
```

```
## Moderately
daily_activity_clean %>%
    summarize(mean = mean(daily_activity_clean$ModeratelyActiveDistance),
        min = min(daily_activity_clean$ModeratelyActiveDistance),
        max = max(daily_activity_clean$ModeratelyActiveDistance))
```

```
## # A tibble: 1 x 3
##    mean   min   max
##   <dbl> <dbl> <dbl>
## 1 0.570     0  6.48
```

```
## VeryActive
daily_activity_clean %>%
    summarize(mean = mean(daily_activity_clean$VeryActiveDistance),
        min = min(daily_activity_clean$VeryActiveDistance), max = max(daily_activity_clean$VeryActiveDi
```

```
## # A tibble: 1 x 3
##    mean   min   max
##   <dbl> <dbl> <dbl>
## 1  1.51     0  21.9
```

Based on the mean, min and max of the different levels, a more suitable cluster for distance/kilometres is created;

- rarely activity

- lightly activity
- moderately activity
- very active

0-3.499 kilometers=> rarely active

3.5-6.499 kilometers=> lightly active

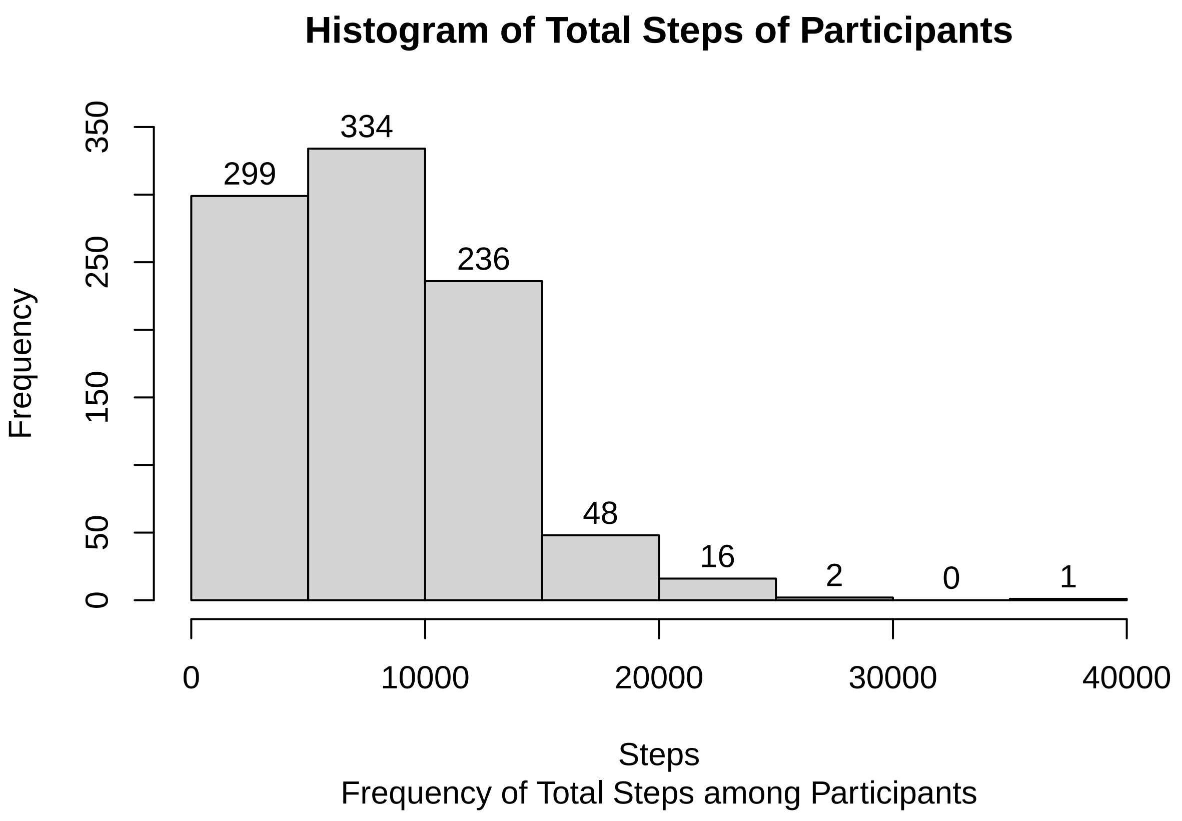
6.5-<=10 kilometers=> active

10->=10 kilometers=> very active

The clustering has been chosen because the histogram indicated that the vast majority of kilometres have been in the range between 0-10 kilometres therefore everything above can be considered very active.

Total Steps

```r
total_steps <- daily_activity_clean %>%
    select(TotalSteps)

total_steps_da <- as.numeric(unlist(total_steps))

hist(total_steps_da, labels = TRUE, xlab = "Steps", ylim = c(0,
    350), main = "Histogram of Total Steps of Participants",
    sub = "Frequency of Total Steps among Participants")
```



Also, the statistical proof that we are dealing with a right-skewed distribution

```
total_steps %>%
    summarize(mean = mean(total_steps_da), median = median(total_steps_da),
        mode = mode(total_steps_da))
```

```
## # A tibble: 1 x 3
##    mean median  mode
##   <dbl>  <dbl> <dbl>
## 1 7671.   7441     0
```

```
total_steps %>%
    summarize(mean = mean(total_steps_da), min = min(total_steps_da),
        max = max(total_steps_da))
```

```
## # A tibble: 1 x 3
##    mean   min   max
##   <dbl> <dbl> <dbl>
## 1 7671.     0 36019
```

Knowing that the data of total steps are right-skewed indicates that the majority of steps are on the left side. The mean, min and max will allow for creating a representative cluster:

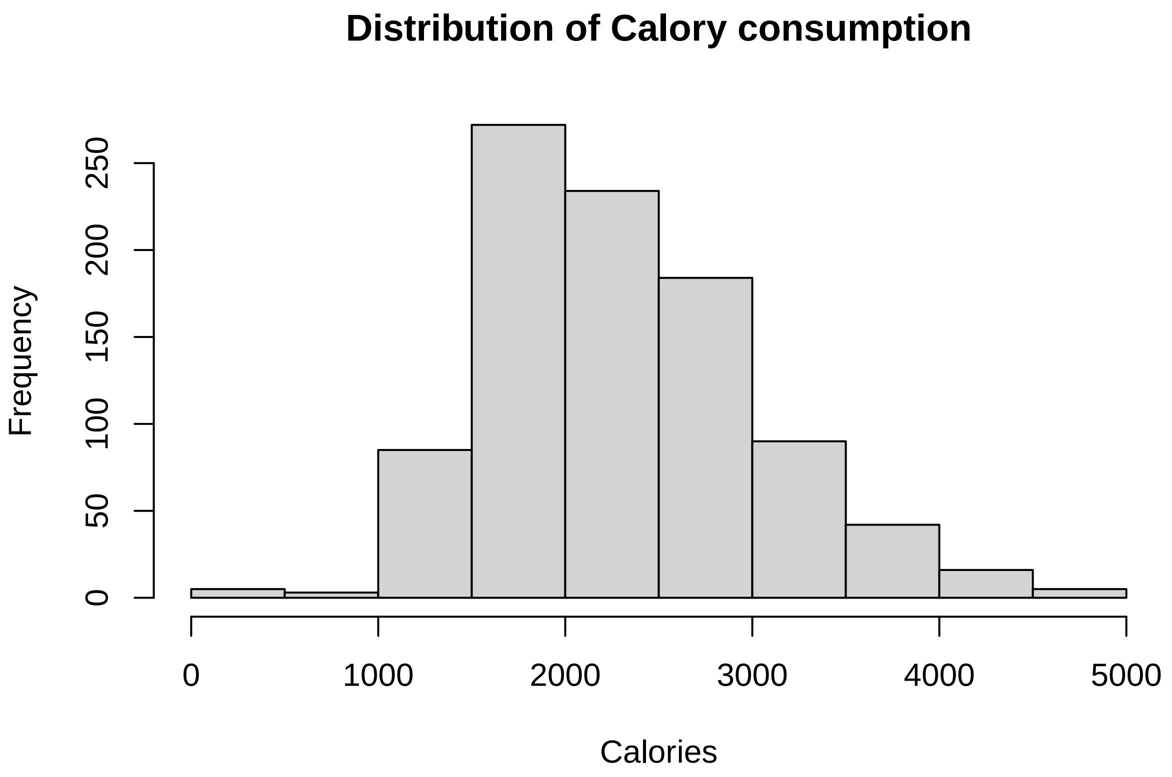
- 0-4.999 steps=> rarely active
- 5.000-7.999 steps=> lightly_active
- 8.000-12.000 steps=> active
- 12.000->= 12.000 steps=> very active

Calories

```
daily_activity_clean %>%
    summarize(mean = mean(daily_activity_clean$Calories), min = min(daily_activity_clean$Calories),
        max = max(daily_activity_clean$Calories))
```

```
## # A tibble: 1 x 3
##    mean   min   max
##   <dbl> <dbl> <dbl>
## 1 2313.    52  4900
```

```
hist(daily_activity_clean$Calories, lables = TRUE, main = "Distribution of Calory consumption",
    xlab = "Calories")
```



There is no information about how many females and males participated; therefore, it is not recommended to draw any conclusions from the data set regarding on how many calories need to be consumed, but it can be helpful to have this data available when we compare our data (calorie consumption of our users) with the current data set.

Hourly breakdown of kilometres, steps and calories

The following paragraph shows in which hours of the day the highest level of steps, calories and activity levels were archived. Worth to mention, the day has been separated in AM and PM. The decision was made to get a better overview on how the participants behaved during the noon and afternoon period.
First, it is important to separate the ActicityHour, to make it easier to visualize the calorie consume over a day

Calory consumption in the AM period

```
hours_12am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "12:00:00 AM"))
avg_12am <- mean(hours_12am$Calories)

hours_1am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "1:00:00 AM"))
avg_1am <- mean(hours_1am$Calories)

hours_2am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "2:00:00 AM"))
avg_2am <- mean(hours_2am$Calories)

hours_3am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "3:00:00 AM"))
```

```r
avg_3am <- mean(hours_3am$Calories)

hours_4am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "4:00:00 AM"))
avg_4am <- mean(hours_4am$Calories)

hours_5am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "5:00:00 AM"))
avg_5am <- mean(hours_5am$Calories)

hours_6am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "6:00:00 AM"))
avg_6am <- mean(hours_6am$Calories)

hours_7am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "7:00:00 AM"))
avg_7am <- mean(hours_7am$Calories)

hours_8am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "8:00:00 AM"))
avg_8am <- mean(hours_8am$Calories)

hours_9am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "9:00:00 AM"))
avg_9am <- mean(hours_9am$Calories)

hours_10am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "10:00:00 AM"))
avg_10am <- mean(hours_10am$Calories)

hours_11am <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "11:00:00 AM"))
avg_11am <- mean(hours_11am$Calories)

time_am <- c("12:00:00", "1:00:00", "2:00:00", "3:00:00", "4:00:00",
    "5:00:00", "6:00:00", "7:00:00", "8:00:00", "9:00:00", "10:00:00",
    "11:00:00")
avg_am <- c(avg_12am, avg_1am, avg_2am, avg_3am, avg_4am, avg_5am,
    avg_6am, avg_7am, avg_8am, avg_9am, avg_10am, avg_11am)
avg_am <- sort(avg_am)
hours_am <- data.frame(time_am, avg_am)

hours_am
```

```
##     time_am    avg_am
## 1  12:00:00  67.53805
## 2   1:00:00  68.26180
## 3   2:00:00  70.49652
## 4   3:00:00  71.80514
## 5   4:00:00  81.70815
## 6   5:00:00  86.99678
## 7   6:00:00  89.92204
## 8   7:00:00  94.47798
## 9   8:00:00 103.33727
## 10  9:00:00 106.14286
## 11 10:00:00 109.80690
## 12 11:00:00 110.46071
```

The table shows that the peak is at 11 AM. It is also observable that in the time period from 8 AM-11 AM, the participants consumed the most calories in the AM period.

Calory consumption in the PM period

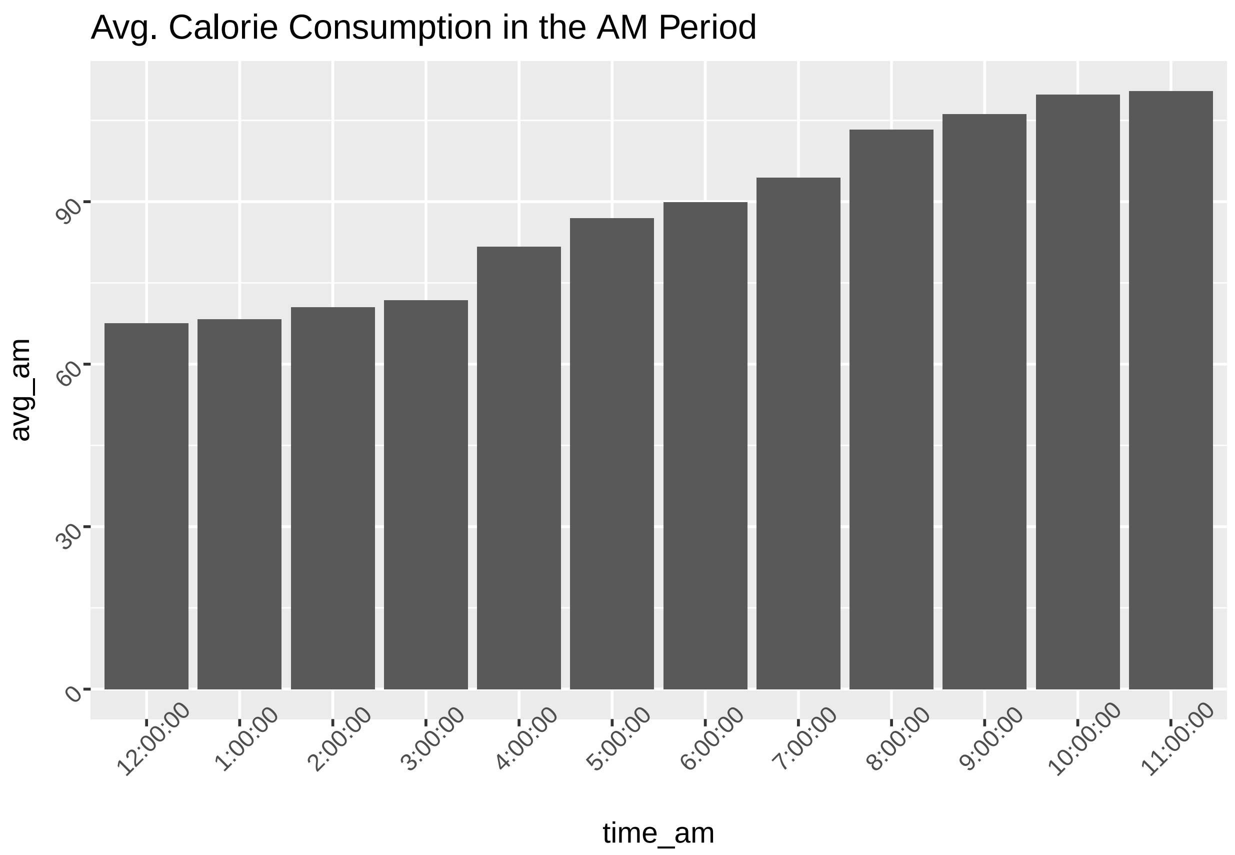
```
hours_12pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "12:00:00 PM"))
avg_12pm <- mean(hours_12pm$Calories)

hours_1pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "1:00:00 PM"))
avg_1pm <- mean(hours_1pm$Calories)

hours_2pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "2:00:00 PM"))
avg_2pm <- mean(hours_2pm$Calories)

hours_3pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "3:00:00 PM"))
avg_3pm <- mean(hours_3pm$Calories)

hours_4pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "4:00:00 PM"))
avg_4pm <- mean(hours_4pm$Calories)

hours_5pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "5:00:00 PM"))
avg_5pm <- mean(hours_5pm$Calories)

hours_6pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "6:00:00 PM"))
avg_6pm <- mean(hours_6pm$Calories)

hours_7pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "7:00:00 PM"))
avg_7pm <- mean(hours_7pm$Calories)

hours_8pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "8:00:00 PM"))
avg_8pm <- mean(hours_8pm$Calories)

hours_9pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "9:00:00 PM"))
avg_9pm <- mean(hours_9pm$Calories)

hours_10pm <- hourly_calories_clean %>%
    filter(str_detect(ActivityHour, "10:00:00 PM"))
avg_10pm <- mean(hours_10pm$Calories)

hours_11pm <- hourly_calories_clean %>%
```

```r
    filter(str_detect(ActivityHour, "11:00:00 PM"))
avg_11pm <- mean(hours_11pm$Calories)

time_pm <- c("12:00:00", "01:00:00", "02:00:00", "03:00:00",
    "04:00:00", "05:00:00", "06:00:00", "07:00:00", "08:00:00",
    "09:00:00", "10:00:00", "11:00:00")
avg_pm <- c(avg_12pm, avg_1pm, avg_2pm, avg_3pm, avg_4pm, avg_5pm,
    avg_6pm, avg_7pm, avg_8pm, avg_9pm, avg_10pm, avg_11pm)

hours_pm <- data.frame(time_pm, avg_pm)

hours_pm
```

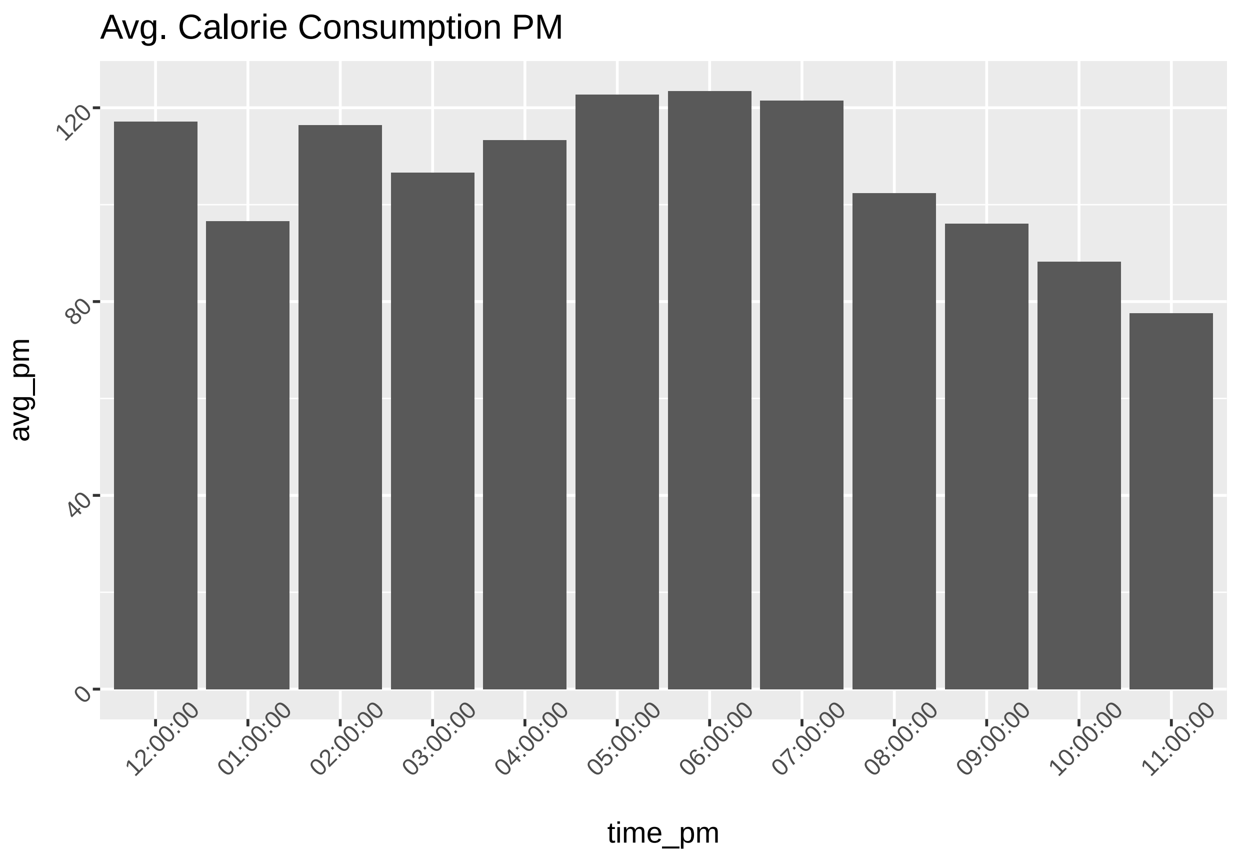
```
##     time_pm    avg_pm
## 1  12:00:00 117.19740
## 2  01:00:00  96.63761
## 3  02:00:00 116.46555
## 4  03:00:00 106.63716
## 5  04:00:00 113.32745
## 6  05:00:00 122.75276
## 7  06:00:00 123.49227
## 8  07:00:00 121.48455
## 9  08:00:00 102.35762
## 10 09:00:00  96.05635
## 11 10:00:00  88.26549
## 12 11:00:00  77.59358
```

To better understanding what these numbers mean, following graphs have been generated, which display the calorie consumption behavior in the AM and PM period.

```r
## AM
calories_plot_am <- ggplot(hours_am, aes(x = time_am, y = avg_am)) +
    geom_bar(stat = "identity") + labs(title = "Avg. Calorie Consumption in the AM Period") +
    theme(axis.text = element_text(angle = 45))

calories_plot_am + scale_x_discrete(limits = time_am)
```



```
## PM
calories_plot_pm <- ggplot(hours_pm, aes(x = time_pm, y = avg_pm)) +
    geom_bar(stat = "identity") + labs(title = "Avg. Calorie Consumption PM") +
    theme(axis.text = element_text(angle = 45))

calories_plot_pm + scale_x_discrete(limits = time_pm)
```



The graphs show that the calorie consumption is constantly increasing from 4 AM to 11 AM, which reaches its maximum for the AM period at 11 AM. In the PM period it is observable at around 12 PM, 2 PM and 5 PM-9 PM that the participants consumed the most calories. Next, the steps and distance walked will be observed.

Steps accumulated in the AM period

```
steps_12am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "12:00:00 AM"))
step_avg_12am <- mean(steps_12am$StepTotal)

steps_1am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "1:00:00 AM"))
step_avg_1am <- mean(steps_1am$StepTotal)

steps_2am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "2:00:00 AM"))
step_avg_2am <- mean(steps_2am$StepTotal)

steps_3am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "3:00:00 AM"))
step_avg_3am <- mean(steps_3am$StepTotal)

steps_4am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "4:00:00 AM"))
step_avg_4am <- mean(steps_4am$StepTotal)
```

```r
steps_5am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "5:00:00 AM"))
step_avg_5am <- mean(steps_5am$StepTotal)

steps_6am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "6:00:00 AM"))
step_avg_6am <- mean(steps_6am$StepTotal)

steps_7am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "7:00:00 AM"))
step_avg_7am <- mean(steps_7am$StepTotal)

steps_8am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "8:00:00 AM"))
step_avg_8am <- mean(steps_8am$StepTotal)

steps_9am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "9:00:00 AM"))
step_avg_9am <- mean(steps_9am$StepTotal)

steps_10am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "10:00:00 AM"))
step_avg_10am <- mean(steps_10am$StepTotal)

steps_11am <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "11:00:00 AM"))
step_avg_11am <- mean(steps_11am$StepTotal)

steps_hours_am <- c("12:00:00 AM", "1:00:00 AM", "2:00:00 AM",
    "3:00:00 AM", "4:00:00 AM", "5:00:00 AM", "6:00:00 AM", "7:00:00 AM",
    "8:00:00 AM", "9:00:00 AM", "10:00:00 AM", "11:00:00 AM")
steps_avg_am <- c(step_avg_12am, step_avg_1am, step_avg_2am,
    step_avg_3am, step_avg_4am, step_avg_5am, step_avg_6am, step_avg_7am,
    step_avg_8am, step_avg_9am, step_avg_10am, step_avg_11am)

steps_am_df <- data.frame(steps_hours_am, steps_avg_am)

steps_am_df
```

```
##    steps_hours_am steps_avg_am
## 1     12:00:00 AM    42.188437
## 2      1:00:00 AM   239.295161
## 3      2:00:00 AM    29.656133
## 4      3:00:00 AM     6.426581
## 5      4:00:00 AM    12.699571
## 6      5:00:00 AM    43.869099
## 7      6:00:00 AM   178.508056
## 8      7:00:00 AM   306.049409
## 9      8:00:00 AM   427.544576
## 10     9:00:00 AM   433.301826
## 11    10:00:00 AM   481.665231
## 12    11:00:00 AM   456.886731
```

The people tend to be more active in the time period from 7 AM-11 AM. An odd observation is at 2 AM, with a relatively high average. This can be due to the small sample size.

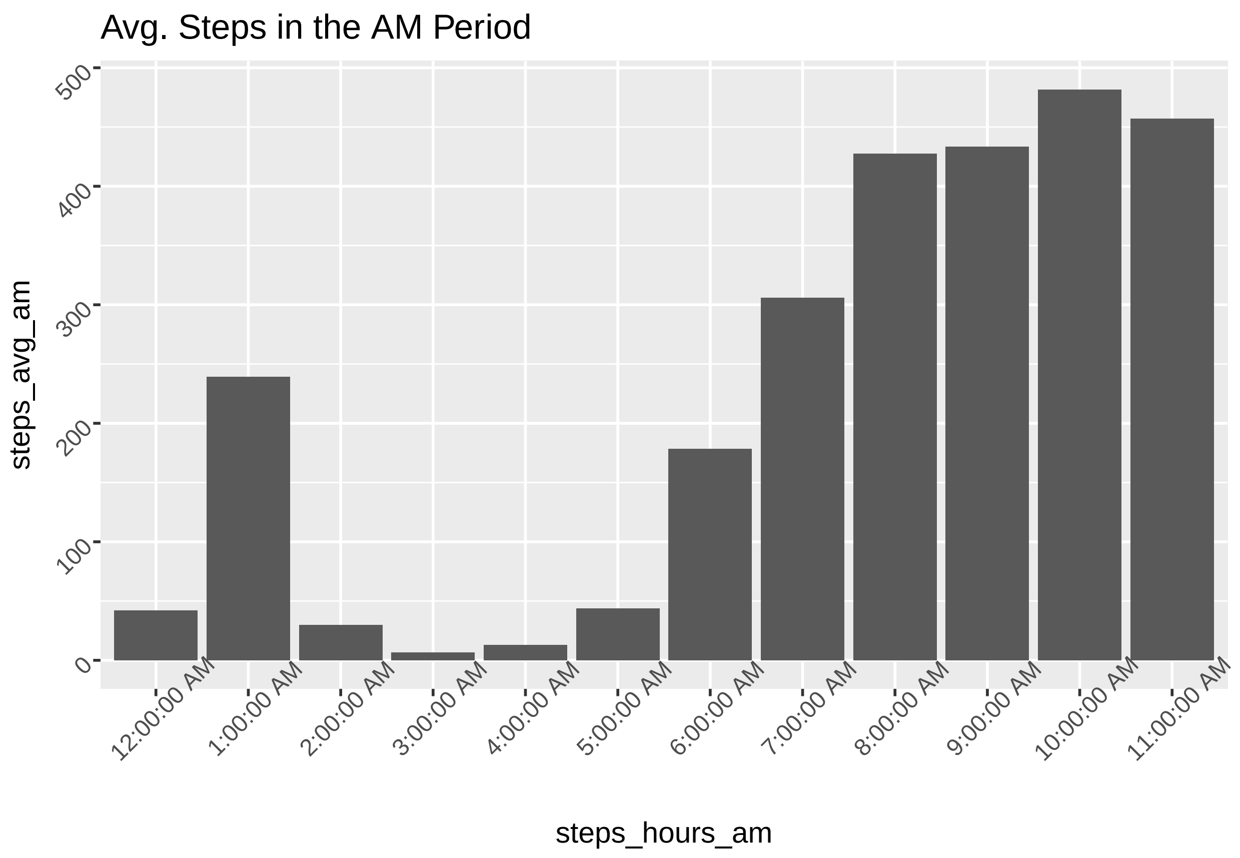
Steps accumulated in the PM period

```r
steps_12pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "12:00:00 PM"))
step_avg_12pm <- mean(steps_12pm$StepTotal)

steps_1pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "1:00:00 PM"))
step_avg_1pm <- mean(steps_1pm$StepTotal)

steps_2pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "2:00:00 PM"))
step_avg_2pm <- mean(steps_2pm$StepTotal)

steps_3pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "3:00:00 PM"))
step_avg_3pm <- mean(steps_3pm$StepTotal)

steps_4pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "4:00:00 PM"))
step_avg_4pm <- mean(steps_4pm$StepTotal)

steps_5pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "5:00:00 PM"))
step_avg_5pm <- mean(steps_5pm$StepTotal)

steps_6pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "6:00:00 PM"))
step_avg_6pm <- mean(steps_6pm$StepTotal)

steps_7pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "7:00:00 PM"))
step_avg_7pm <- mean(steps_7pm$StepTotal)

steps_8pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "8:00:00 PM"))
step_avg_8pm <- mean(steps_8pm$StepTotal)

steps_9pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "9:00:00 PM"))
step_avg_9pm <- mean(steps_9pm$StepTotal)

steps_10pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "10:00:00 PM"))
step_avg_10pm <- mean(steps_10pm$StepTotal)

steps_11pm <- hourly_steps_clean %>%
    filter(str_detect(ActivityHour, "11:00:00 PM"))
step_avg_11pm <- mean(steps_11pm$StepTotal)

steps_hours_pm <- c("12:00:00 PM", "1:00:00 PM", "2:00:00 PM",
    "3:00:00 PM", "4:00:00 PM", "5:00:00 PM", "6:00:00 PM", "7:00:00 PM",
    "8:00:00 PM", "9:00:00 PM", "10:00:00 PM", "11:00:00 PM")
steps_avg_pm <- c(step_avg_12pm, step_avg_1pm, step_avg_2pm,
    step_avg_3pm, step_avg_4pm, step_avg_5pm, step_avg_6pm, step_avg_7pm,
    step_avg_8pm, step_avg_9pm, step_avg_10pm, step_avg_11pm)

steps_pm_df <- data.frame(steps_hours_pm, steps_avg_pm)

steps_pm_df
```

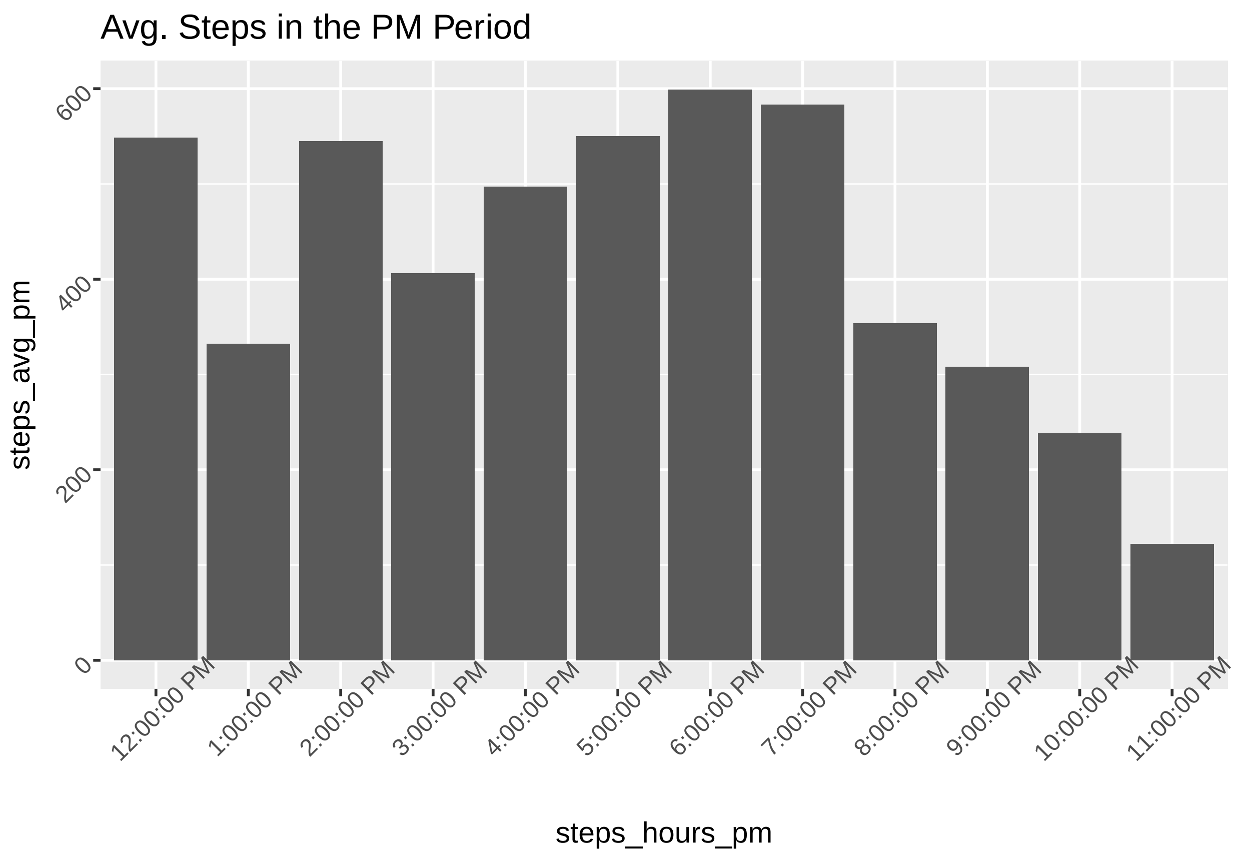
```
##    steps_hours_pm steps_avg_pm
## 1     12:00:00 PM     548.6421
## 2      1:00:00 PM     331.9660
## 3      2:00:00 PM     544.5800
## 4      3:00:00 PM     406.3191
## 5      4:00:00 PM     496.8456
## 6      5:00:00 PM     550.2329
## 7      6:00:00 PM     599.1700
## 8      7:00:00 PM     583.3907
## 9      8:00:00 PM     353.9051
## 10     9:00:00 PM     308.1381
## 11    10:00:00 PM     237.9878
## 12    11:00:00 PM     122.1329
```

The most steps in the PM period are accumulated at 12 PM, 2 PM and 5 PM-7 PM.
These findings are presented in the following graphs.

```
steps_plot_am <- ggplot(steps_am_df, aes(x = steps_hours_am,
    y = steps_avg_am)) + geom_bar(stat = "identity") + labs(title = "Avg. Steps in the AM Period") +
    theme(axis.text = element_text(angle = 45))

steps_plot_am + scale_x_discrete(limits = steps_hours_am)
```



```r
steps_plot_pm <- ggplot(steps_pm_df, aes(x = steps_hours_pm,
    y = steps_avg_pm)) + geom_bar(stat = "identity") + labs(title = "Avg. Steps in the PM Period") +
    theme(axis.text = element_text(angle = 45))

steps_plot_pm + scale_x_discrete(limits = steps_hours_pm)
```



It is a familiar behavior observable, people tend to be more active around 12 PM and 5 PM-7 PM. There could be a trend noticeable, which will be analyzed later on.

Intensities

The data set "hourly_intensities_clean" provides information about how many minutes in each hour were spent doing activities

```
hourly_intensities_clean
```

```
## # A tibble: 22,099 x 4
##            Id ActivityHour          TotalIntensity AverageIntensity
##         <dbl> <chr>                          <dbl>            <dbl>
##  1 1503960366 4/12/2016 12:00:00 AM             20            0.333
##  2 1503960366 4/12/2016 1:00:00 AM               8            0.133
##  3 1503960366 4/12/2016 2:00:00 AM               7            0.117
##  4 1503960366 4/12/2016 3:00:00 AM               0            0
##  5 1503960366 4/12/2016 4:00:00 AM               0            0
##  6 1503960366 4/12/2016 5:00:00 AM               0            0
##  7 1503960366 4/12/2016 6:00:00 AM               0            0
##  8 1503960366 4/12/2016 7:00:00 AM               0            0
##  9 1503960366 4/12/2016 8:00:00 AM              13            0.217
## 10 1503960366 4/12/2016 9:00:00 AM              30            0.5
## # ... with 22,089 more rows
```

```
mean(hourly_intensities_clean$TotalIntensity)
```

```
## [1] 12.03534
```

The data indicates that the participants spent 12 min on average in every hour of the day nonetheless, it would be helpful to understand how the active minutes are distributed over the day.

Distribution of Active Minutes during the AM period

```r
inten_12am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "12:00:00 AM"))
inten_avg_12am <- mean(inten_12am$TotalIntensity)

inten_1am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "1:00:00 AM"))
inten_avg_1am <- mean(inten_1am$TotalIntensity)

inten_2am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "2:00:00 AM"))
inten_avg_2am <- mean(inten_2am$TotalIntensity)

inten_3am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "3:00:00 AM"))
inten_avg_3am <- mean(inten_3am$TotalIntensity)

inten_4am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "4:00:00 AM"))
inten_avg_4am <- mean(inten_4am$TotalIntensity)

inten_5am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "5:00:00 AM"))
inten_avg_5am <- mean(inten_5am$TotalIntensity)

inten_6am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "6:00:00 AM"))
inten_avg_6am <- mean(inten_6am$TotalIntensity)

inten_7am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "7:00:00 AM"))
inten_avg_7am <- mean(inten_7am$TotalIntensity)

inten_8am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "8:00:00 AM"))
inten_avg_8am <- mean(inten_8am$TotalIntensity)

inten_9am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "9:00:00 AM"))
inten_avg_9am <- mean(inten_9am$TotalIntensity)

inten_10am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "10:00:00 AM"))
inten_avg_10am <- mean(inten_10am$TotalIntensity)

inten_11am <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "11:00:00 AM"))
inten_avg_11am <- mean(inten_11am$TotalIntensity)

inten_hours_am <- c("12:00:00 AM", "1:00:00 AM", "2:00:00 AM",
    "3:00:00 AM", "4:00:00 AM", "5:00:00 AM", "6:00:00 AM", "7:00:00 AM",
    "8:00:00 AM", "9:00:00 AM", "10:00:00 AM", "11:00:00 AM")
inten_avg_am <- c(inten_avg_12am, inten_avg_1am, inten_avg_2am,
    inten_avg_3am, inten_avg_4am, inten_avg_5am, inten_avg_6am,
    inten_avg_7am, inten_avg_8am, inten_avg_9am, inten_avg_10am,
    inten_avg_11am)

inten_am_df <- data.frame(inten_hours_am, inten_avg_am)

inten_am_df
```

```
##    inten_hours_am inten_avg_am
## 1     12:00:00 AM    2.1295503
## 2      1:00:00 AM    9.1451613
## 3      2:00:00 AM    1.5870380
## 4      3:00:00 AM    0.4437299
## 5      4:00:00 AM    0.6330472
## 6      5:00:00 AM    4.9506438
## 7      6:00:00 AM    7.7712137
## 8      7:00:00 AM   10.7336198
## 9      8:00:00 AM   14.6680988
## 10     9:00:00 AM   15.3877551
## 11    10:00:00 AM   17.6437029
## 12    11:00:00 AM   16.9212513
```

The table shows that the most active hours during the AM period is the slot between 8 AM-11 AM.

PM

```
inten_12pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "12:00:00 PM"))
inten_avg_12pm <- mean(inten_12pm$TotalIntensity)

inten_1pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "1:00:00 PM"))
inten_avg_1pm <- mean(inten_1pm$TotalIntensity)

inten_2pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "2:00:00 PM"))
inten_avg_2pm <- mean(inten_2pm$TotalIntensity)

inten_3pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "3:00:00 PM"))
inten_avg_3pm <- mean(inten_3pm$TotalIntensity)

inten_4pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "4:00:00 PM"))
inten_avg_4pm <- mean(inten_4pm$TotalIntensity)

inten_5pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "5:00:00 PM"))
inten_avg_5pm <- mean(inten_5pm$TotalIntensity)

inten_6pm <- hourly_intensities_clean %>%
```

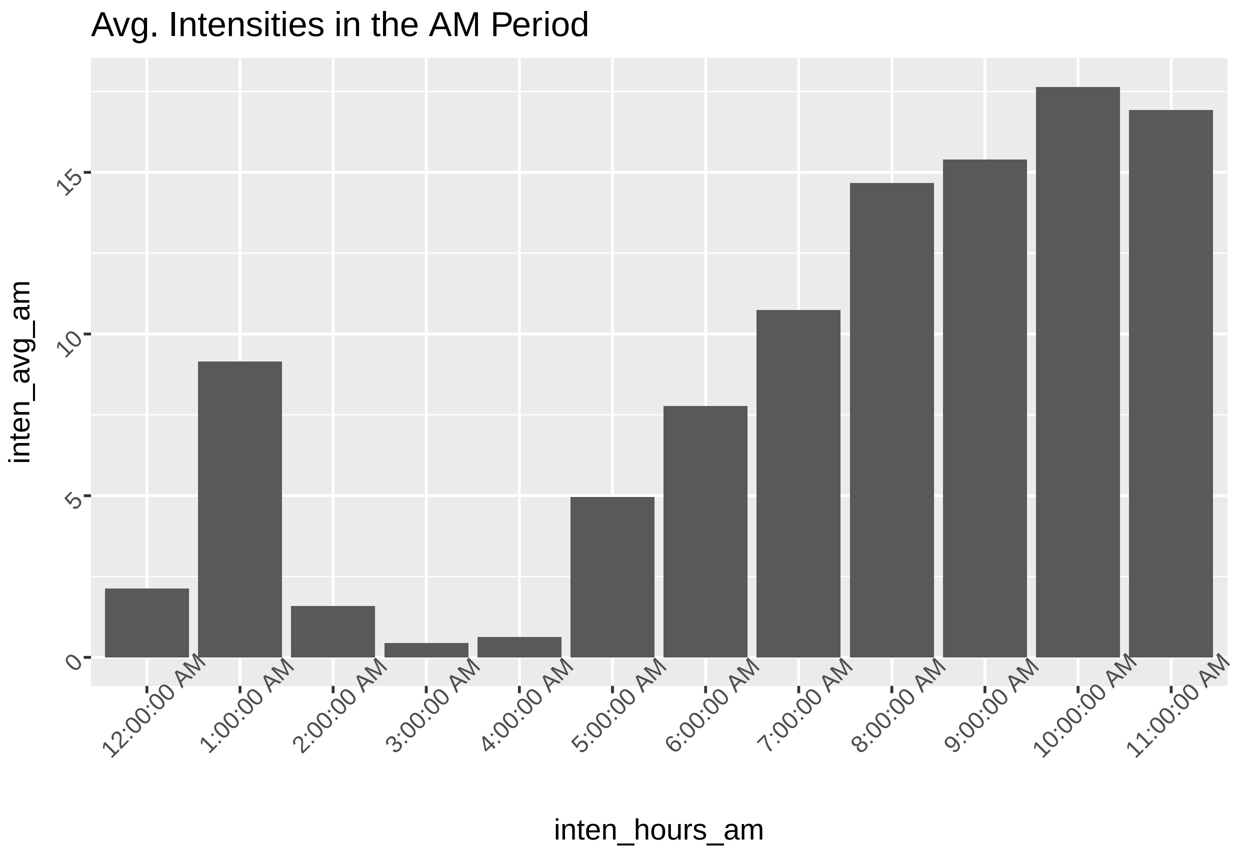
```r
    filter(str_detect(ActivityHour, "6:00:00 PM"))
inten_avg_6pm <- mean(inten_6pm$TotalIntensity)

inten_7pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "7:00:00 PM"))
inten_avg_7pm <- mean(inten_7pm$TotalIntensity)

inten_8pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "8:00:00 PM"))
inten_avg_8pm <- mean(inten_8pm$TotalIntensity)

inten_9pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "9:00:00 PM"))
inten_avg_9pm <- mean(inten_9pm$TotalIntensity)

inten_10pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "10:00:00 PM"))
inten_avg_10pm <- mean(inten_10pm$TotalIntensity)

inten_11pm <- hourly_intensities_clean %>%
    filter(str_detect(ActivityHour, "11:00:00 PM"))
inten_avg_11pm <- mean(inten_11pm$TotalIntensity)

inten_hours_pm <- c("12:00:00 PM", "1:00:00 PM", "2:00:00 PM",
    "3:00:00 PM", "4:00:00 PM", "5:00:00 PM", "6:00:00 PM", "7:00:00 PM",
    "8:00:00 PM", "9:00:00 PM", "10:00:00 PM", "11:00:00 PM")
inten_avg_pm <- c(inten_avg_12pm, inten_avg_1pm, inten_avg_2pm,
    inten_avg_3pm, inten_avg_4pm, inten_avg_5pm, inten_avg_6pm,
    inten_avg_7pm, inten_avg_8pm, inten_avg_9pm, inten_avg_10pm,
    inten_avg_11pm)

inten_pm_df <- data.frame(inten_hours_pm, inten_avg_pm)

inten_pm_df
```

```
##    inten_hours_pm inten_avg_pm
## 1     12:00:00 PM    19.847072
## 2      1:00:00 PM    11.953947
## 3      2:00:00 PM    19.358112
## 4      3:00:00 PM    15.584699
## 5      4:00:00 PM    17.716648
## 6      5:00:00 PM    21.655629
## 7      6:00:00 PM    21.921634
## 8      7:00:00 PM    21.385210
## 9      8:00:00 PM    14.339956
## 10     9:00:00 PM    12.072928
## 11    10:00:00 PM     9.063053
## 12    11:00:00 PM     4.996678
```

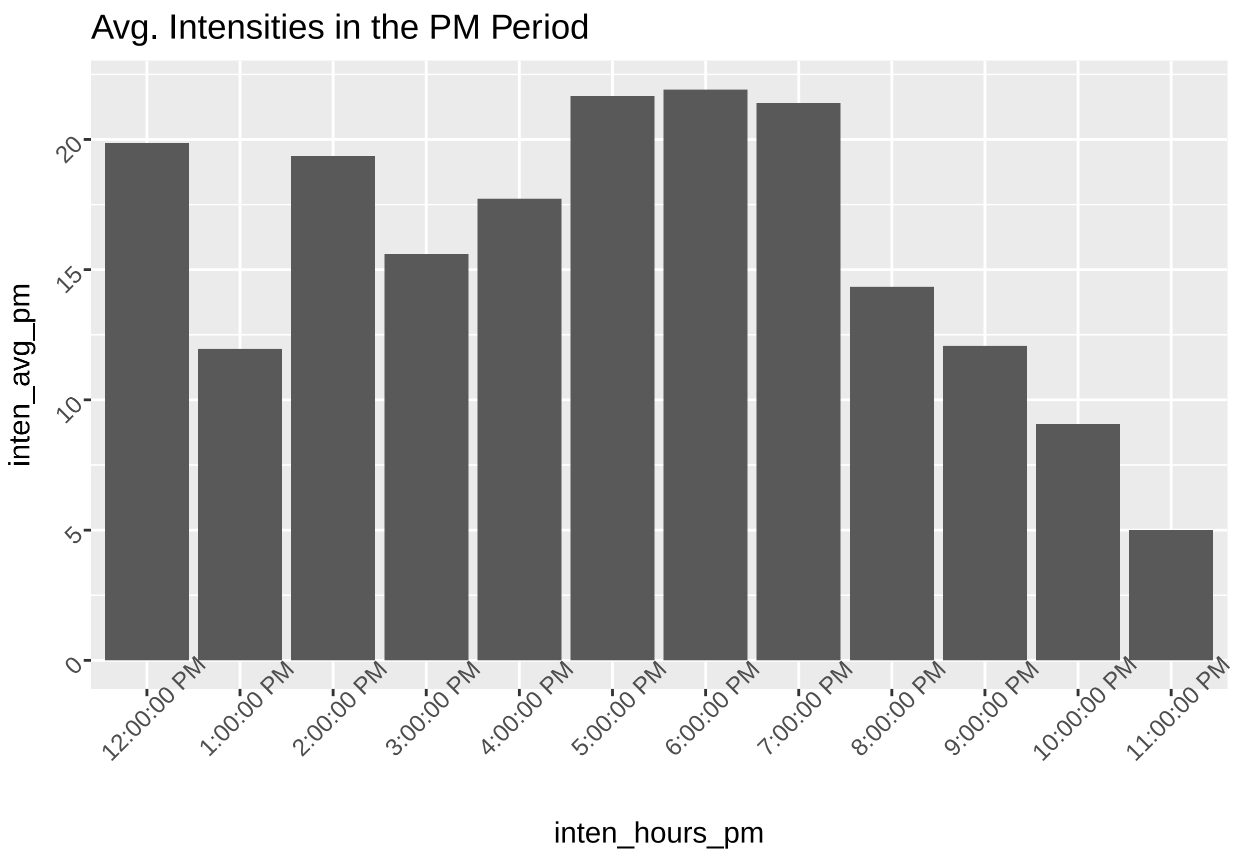
By reading the table, it becomes apparent that we are observing the same behavior again. People become the most active around 5 PM-7 PM.

```r
inten_plot_am <- ggplot(inten_am_df, aes(x = inten_hours_am,
    y = inten_avg_am)) + geom_bar(stat = "identity") + labs(title = "Avg. Intensities in the AM Period"
    theme(axis.text = element_text(angle = 45))
```

```r
inten_plot_am + scale_x_discrete(limits = inten_hours_am)
```



```r
inten_plot_pm <- ggplot(inten_pm_df, aes(x = inten_hours_pm,
    y = inten_avg_pm)) + geom_bar(stat = "identity") + labs(title = "Avg. Intensities in the PM Period")
    theme(axis.text = element_text(angle = 45))

inten_plot_pm + scale_x_discrete(limits = inten_hours_pm)
```



The most active minutes of the day are accumulated during the time period of 5 PM-7 PM. These specific periods have already been observed with the variables.

Sleep

Next, it is also important to get an idea of what the sleeping behavior of the participants is like.

```
daily_sleep_clean
```

```
## # A tibble: 413 x 5
##             Id SleepDay           TotalSleepRecor~ TotalMinutesAsl~ TotalTimeInBed
##          <dbl> <chr>                         <dbl>            <dbl>          <dbl>
##  1 1503960366 4/12/2016 12:00:~                  1              327            346
##  2 1503960366 4/13/2016 12:00:~                  2              384            407
##  3 1503960366 4/15/2016 12:00:~                  1              412            442
##  4 1503960366 4/16/2016 12:00:~                  2              340            367
##  5 1503960366 4/17/2016 12:00:~                  1              700            712
##  6 1503960366 4/19/2016 12:00:~                  1              304            320
##  7 1503960366 4/20/2016 12:00:~                  1              360            377
##  8 1503960366 4/21/2016 12:00:~                  1              325            364
##  9 1503960366 4/23/2016 12:00:~                  1              361            384
## 10 1503960366 4/24/2016 12:00:~                  1              430            449
## # ... with 403 more rows
```

```
avg_time_asleep <- round((mean(daily_sleep_clean$TotalMinutesAsleep)/60))
avg_time_in_bed <- round((mean(daily_sleep_clean$TotalTimeInBed)/60))

avg_time_asleep
```

```
## [1] 7
```

```
avg_time_in_bed
```

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

The average participant slept 7h a day and spent 8h in bed, an assumption would be that participants need 30 min to fall asleep, and it takes them 30 min to get up and leave the bed.

Breaking down sleep, calories, activity, steps and distance on weekdays

The best way to do so is to merge the dailyActivity_clean and sleepDay_clean.

```
act_sleep_df <- merge(x = daily_activity_clean, y = daily_sleep_clean,
    c("Id"))
head(act_sleep_df)
```

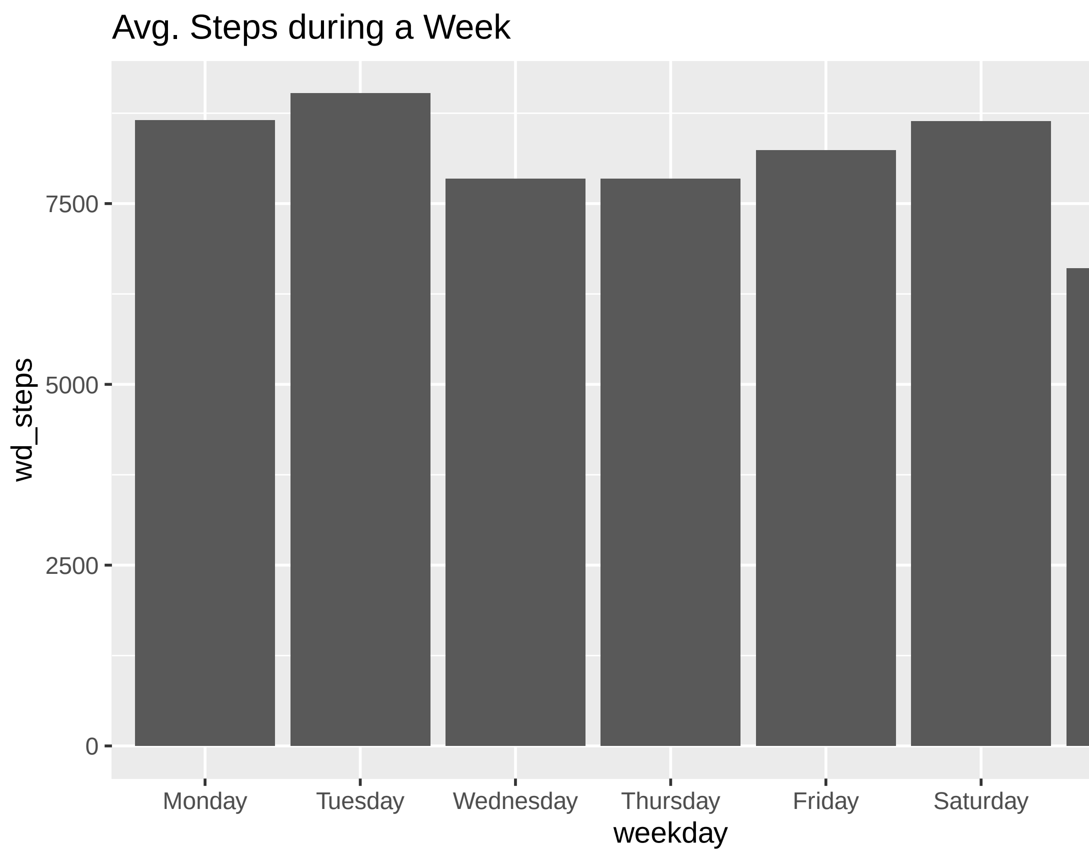
```
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
## 1 1503960366    4/29/2016      11181          7.15            7.15
## 2 1503960366    4/29/2016      11181          7.15            7.15
## 3 1503960366    4/29/2016      11181          7.15            7.15
## 4 1503960366    4/29/2016      11181          7.15            7.15
## 5 1503960366    4/29/2016      11181          7.15            7.15
## 6 1503960366    4/29/2016      11181          7.15            7.15
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               1.06                      0.5
## 2                        0               1.06                      0.5
## 3                        0               1.06                      0.5
## 4                        0               1.06                      0.5
## 5                        0               1.06                      0.5
## 6                        0               1.06                      0.5
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                5.58                       0                16
## 2                5.58                       0                16
## 3                5.58                       0                16
## 4                5.58                       0                16
## 5                5.58                       0                16
## 6                5.58                       0                16
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories
## 1                  12                  243              815     1837
## 2                  12                  243              815     1837
## 3                  12                  243              815     1837
## 4                  12                  243              815     1837
## 5                  12                  243              815     1837
## 6                  12                  243              815     1837
##                SleepDay TotalSleepRecords TotalMinutesAsleep TotalTimeInBed
## 1 4/12/2016 12:00:00 AM                 1                327            346
## 2 4/13/2016 12:00:00 AM                 2                384            407
## 3 4/15/2016 12:00:00 AM                 1                412            442
## 4 4/16/2016 12:00:00 AM                 2                340            367
## 5 4/17/2016 12:00:00 AM                 1                700            712
## 6 4/19/2016 12:00:00 AM                 1                304            320
```

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

weekday_act_sleep_df <- date_act_sleep_df %>%
    mutate(weekday = weekdays(date))

weekday_act_sleep_df$weekday <- ordered(weekday_act_sleep_df$weekday,
    levels = c("Monday", "Tuesday", "Wednesday", "Thursday",
        "Friday", "Saturday", "Sunday"))

weekday_act_sleep_df <- weekday_act_sleep_df %>%
    group_by(weekday) %>%
    summarize(wd_steps = mean(TotalSteps), wd_sleep = mean(TotalMinutesAsleep),
        wd_calories = mean(Calories), wd_distance = mean(TotalDistance))

weekday_act_sleep_df
```

Transforming into weekdays

```
## # A tibble: 7 x 5
##   weekday   wd_steps wd_sleep wd_calories wd_distance
##   <ord>        <dbl>    <dbl>       <dbl>       <dbl>
## 1 Monday       8653.     420.       2387.        6.12
## 2 Tuesday      9022.     419.       2421.        6.35
## 3 Wednesday    7845.     419.       2295.        5.56
## 4 Thursday     7841.     422.       2246.        5.54
## 5 Friday       8237.     419.       2382.        5.78
## 6 Saturday     8639.     419.       2383.        6.09
## 7 Sunday       6600.     420.       2227.        4.72
```

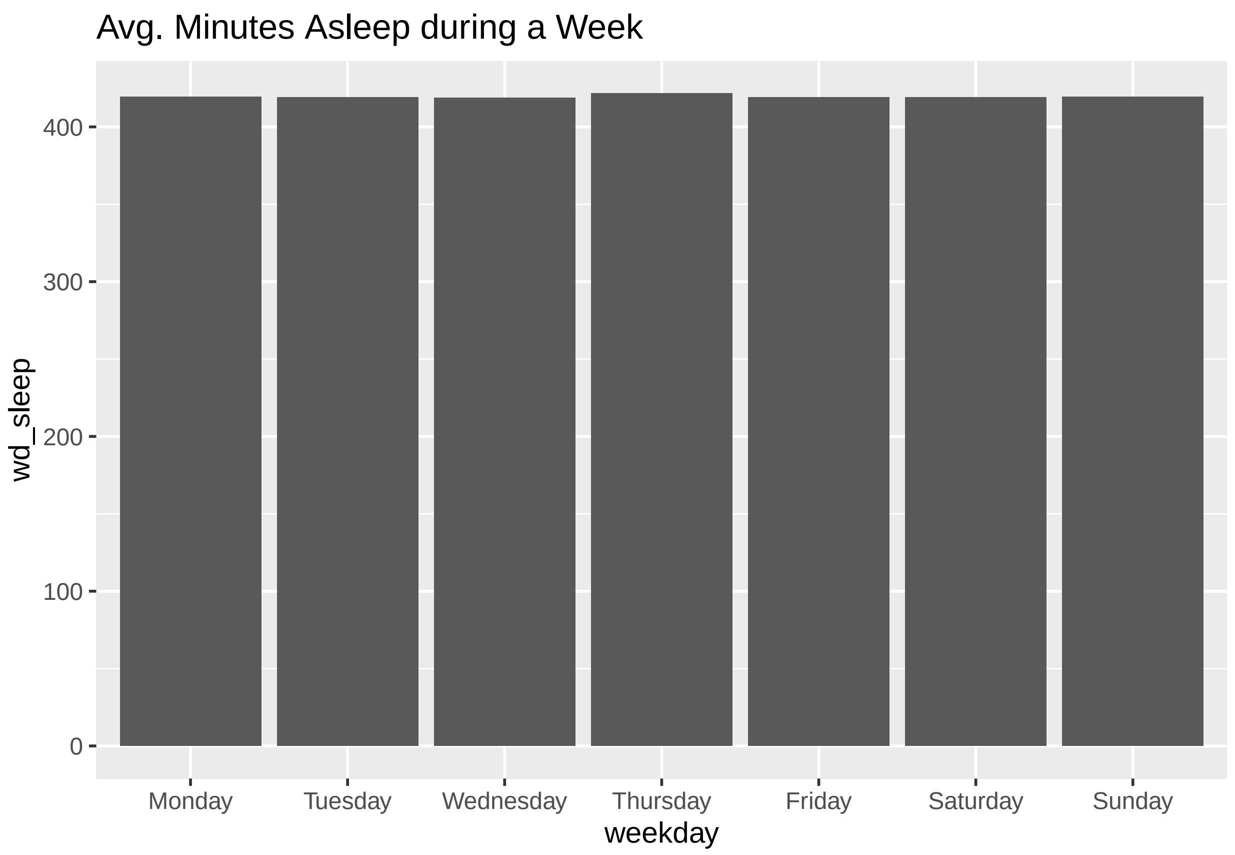
```
ggplot(weekday_act_sleep_df, aes(x = weekday, y = wd_steps)) +
    geom_bar(stat = "identity") + labs(title = "Avg. Steps during a Week")
```



Breaking it down for each

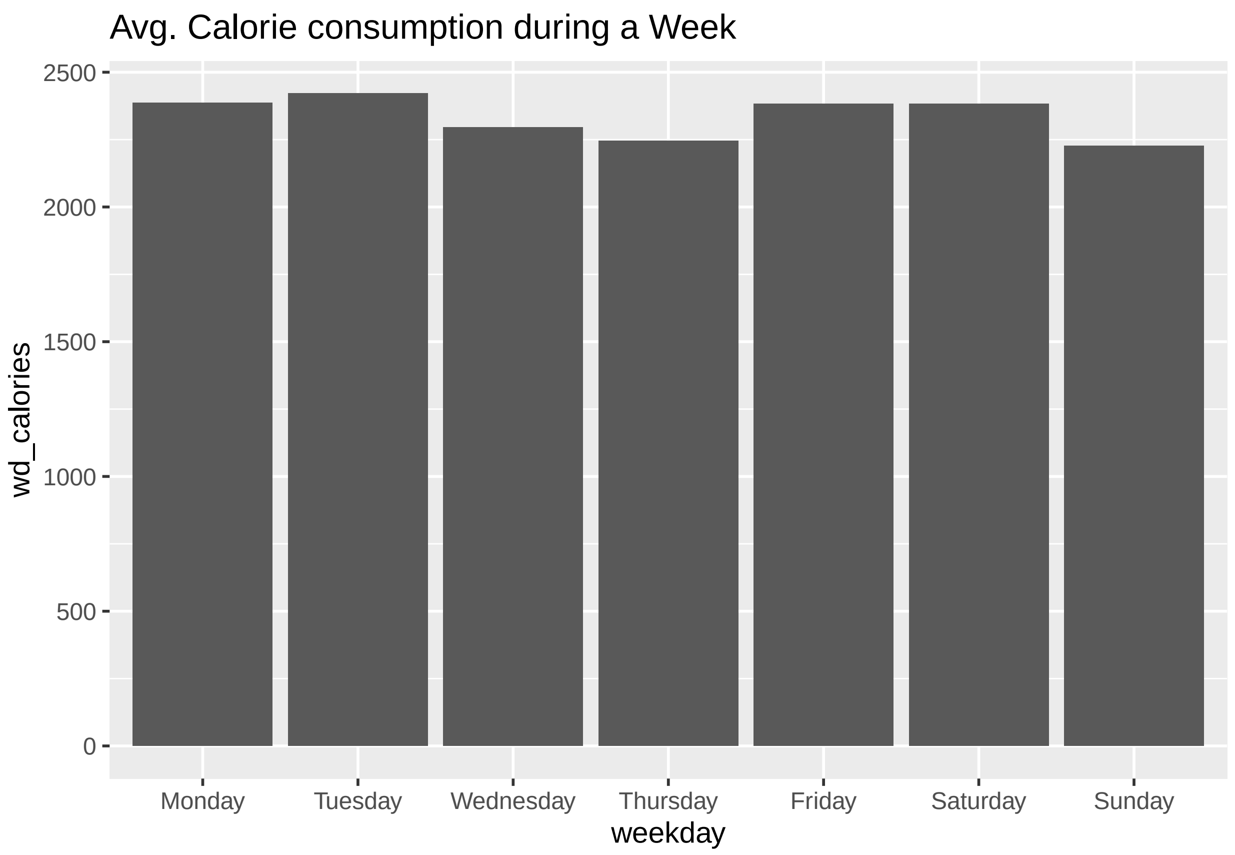
People tend to walk fewer steps on Sundays. The peak days are Monday, Tuesday and Saturday.

```
ggplot(weekday_act_sleep_df, aes(x = weekday, y = wd_sleep)) +
    geom_bar(stat = "identity") + labs(title = "Avg. Minutes Asleep during a Week")
```



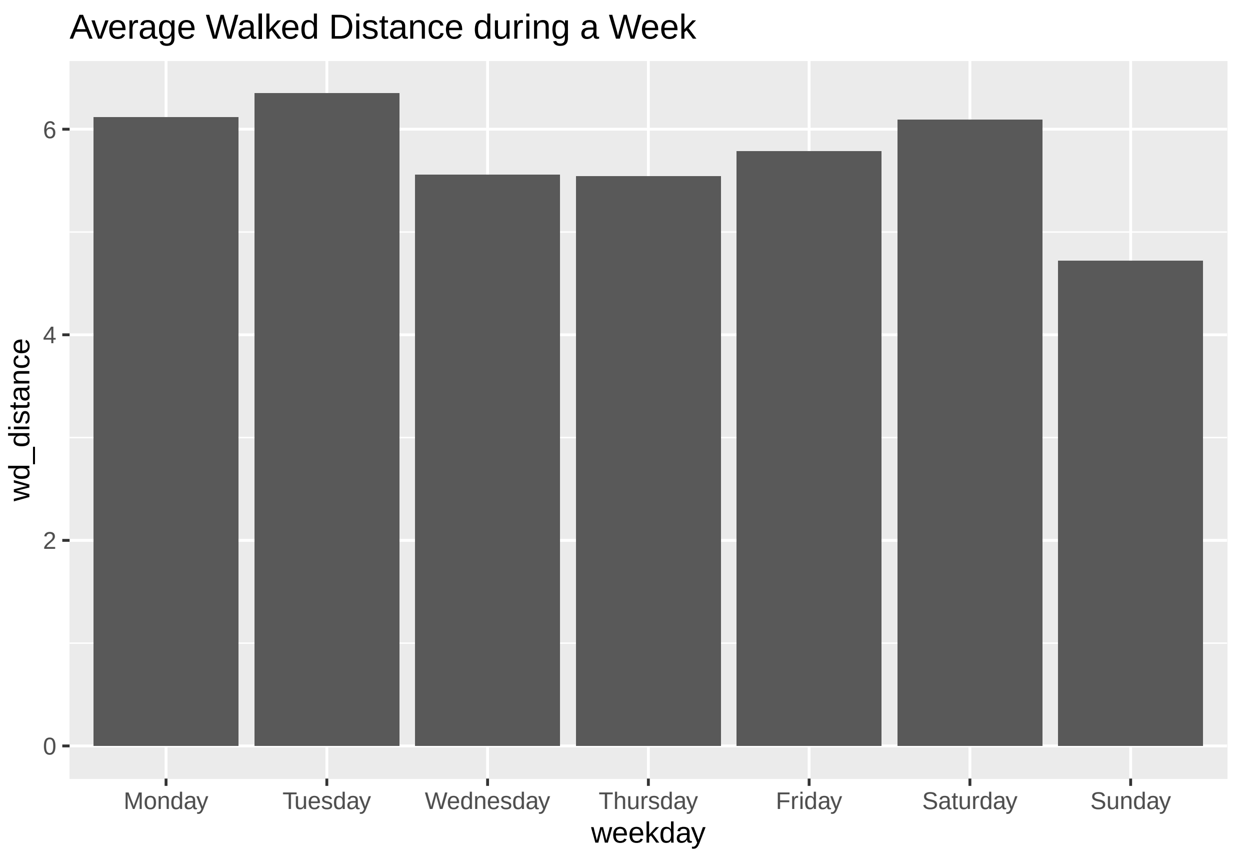
Regarding sleep, the sleeping pattern did not change much over the period of a week (avg. 7h).

```
ggplot(weekday_act_sleep_df, aes(x = weekday, y = wd_calories)) +
    geom_bar(stat = "identity") + labs(title = "Avg. Calorie consumption during a Week")
```



People tend to consume fewer calories on Tuesdays, Wednesdays and Sundays.

```
ggplot(weekday_act_sleep_df, aes(x = weekday, y = wd_distance)) +
    geom_bar(stat = "identity") + labs(title = "Average Walked Distance during a Week")
```



People walk less on Sundays. The most distance walked is on Tuesdays.

Correlation & R-squared

After analyzing the descriptive data and getting a good impression of how a typical smart device user might behave, the following paragraph will analyze observable trends of the data. The correlation and R-squared will do this. A quick review/explanation of what the correlation and R-squared are. The correlation allows analysts to determine if there is a strong relationship between two values. R-squared is used to determine how good a value (independent variable) can be used to predict a dependent variable.

Calories as dependent variable

Total Distance vs Calories

```
dist_cal <- daily_activity_clean %>%
    select(TotalDistance, Calories)

cor(dist_cal)
```

```
##               TotalDistance  Calories
## TotalDistance     1.0000000 0.6427066
## Calories          0.6427066 1.0000000
```

The outcome is a correlation of 0.6427, a moderate correlation level (Mukaka, 2012).

To calculate R-squared it is required to create the function rsq, to calculate R-squared.

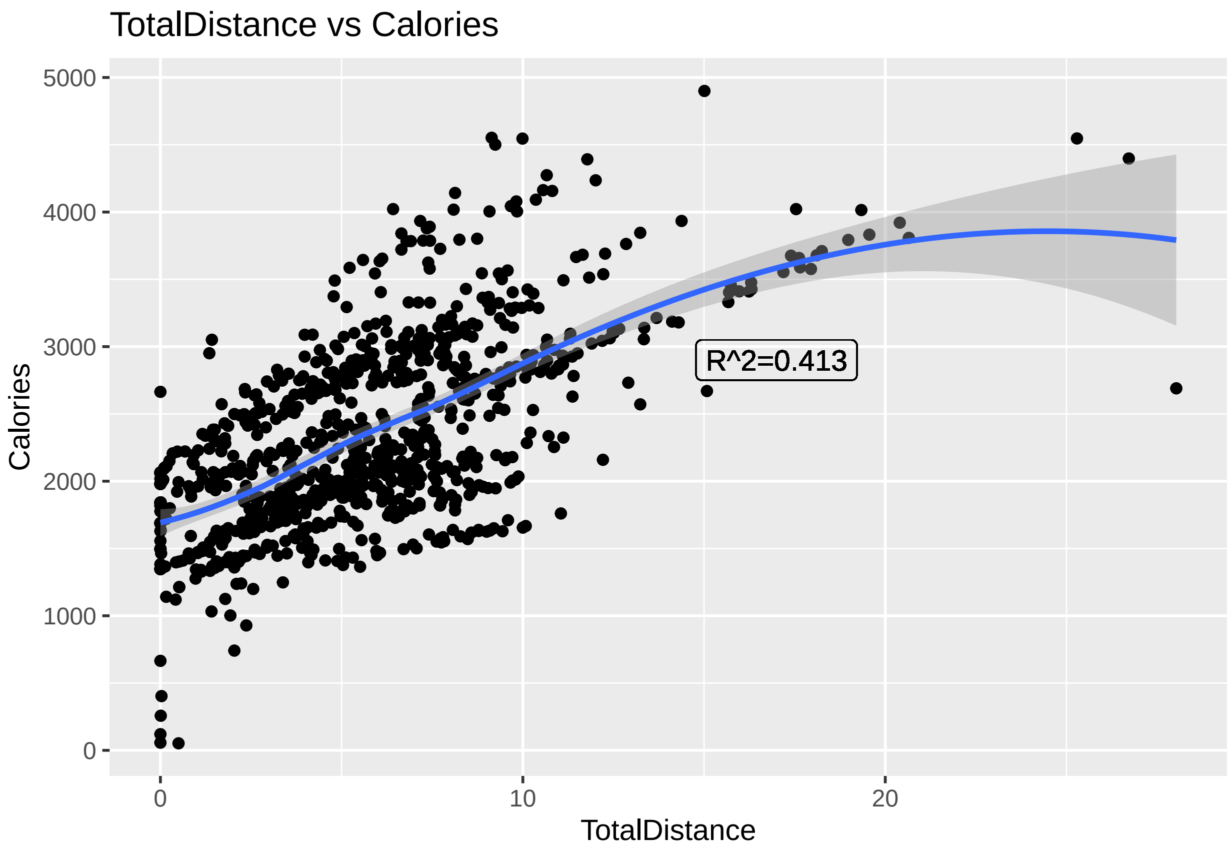
```
rsq <- function(x) cor(x)^2

rsq(dist_cal)
```

```
##               TotalDistance  Calories
## TotalDistance     1.0000000 0.4130718
## Calories          0.4130718 1.0000000
```

R-squared is 0.413, a moderate predictor for the dependent variable calories (Fernando, 2021).

```
ggplot(dist_cal, aes(x = TotalDistance, y = Calories)) + geom_point() +
    geom_smooth() + labs(title = "TotalDistance vs Calories") +
    geom_label(label = "R^2=0.413", x = 17, y = 2900, fill = NA)
```

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



Total Steps vs Calories

```
tost_cal <- daily_activity_clean %>%
    select(TotalSteps, Calories)

cor(tost_cal)
```

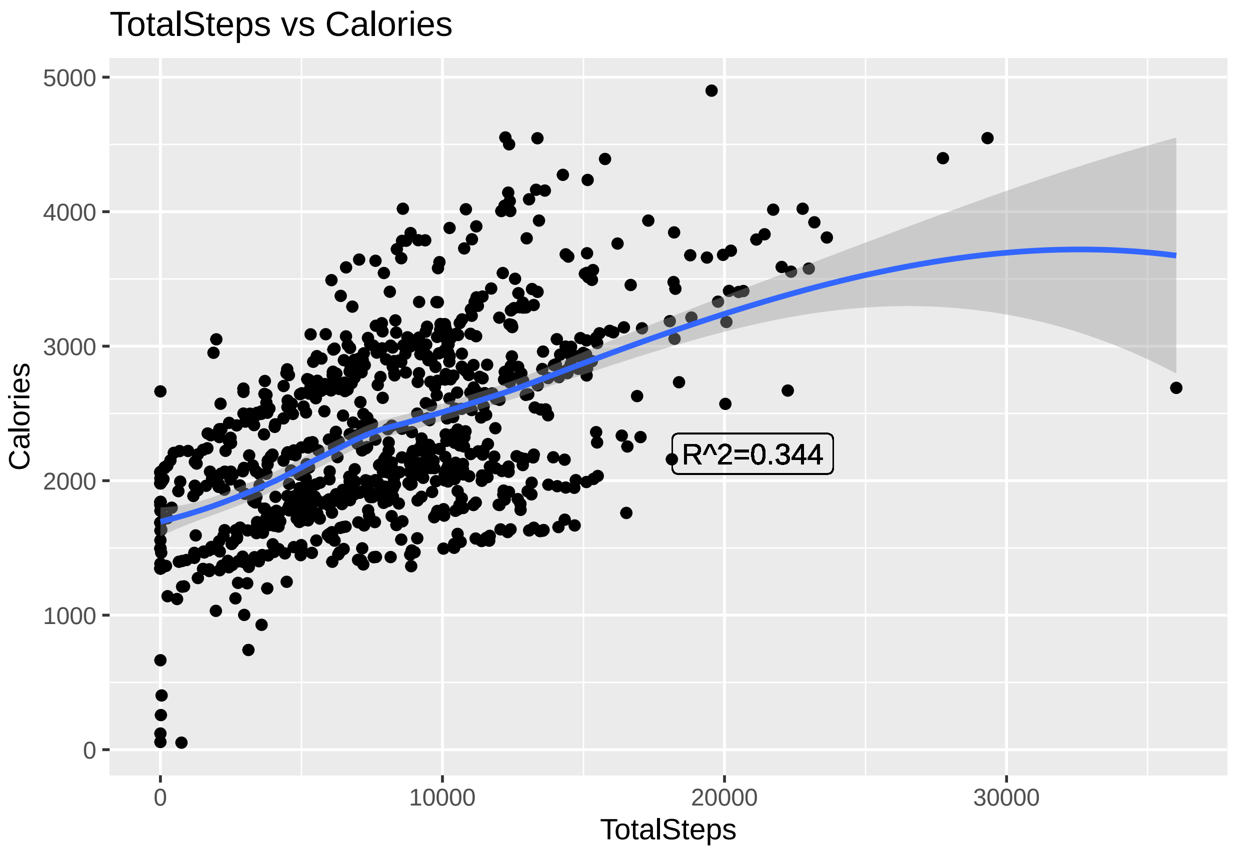
```
##            TotalSteps Calories
## TotalSteps   1.000000 0.586798
## Calories     0.586798 1.000000
```

```
rsq(tost_cal)
```

```
##             TotalSteps  Calories
## TotalSteps  1.0000000 0.3443319
## Calories    0.3443319 1.0000000
```

```
ggplot(tost_cal, aes(x = TotalSteps, y = Calories)) + geom_point() +
    geom_smooth() + labs(title = "TotalSteps vs Calories") +
    geom_label(label = "R^2=0.344", x = 21000, y = 2200, fill = NA)
```

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



The outcome shows that total steps and calories do not have a high correlation, and TotalSteps is not a good predictor for Calories (Mukaka, 2012), (Fernando, 2021).

Different levels of activities vs Calories Next, the correlation between intensity and calories will be measured. It is important to remember that activity is divided into different levels; each level will be analysed.

```
in_s_cal <- daily_activity_clean %>%
    select(SedentaryMinutes, Calories)

cor(in_s_cal)
```

Sedentary Minutes vs Calories

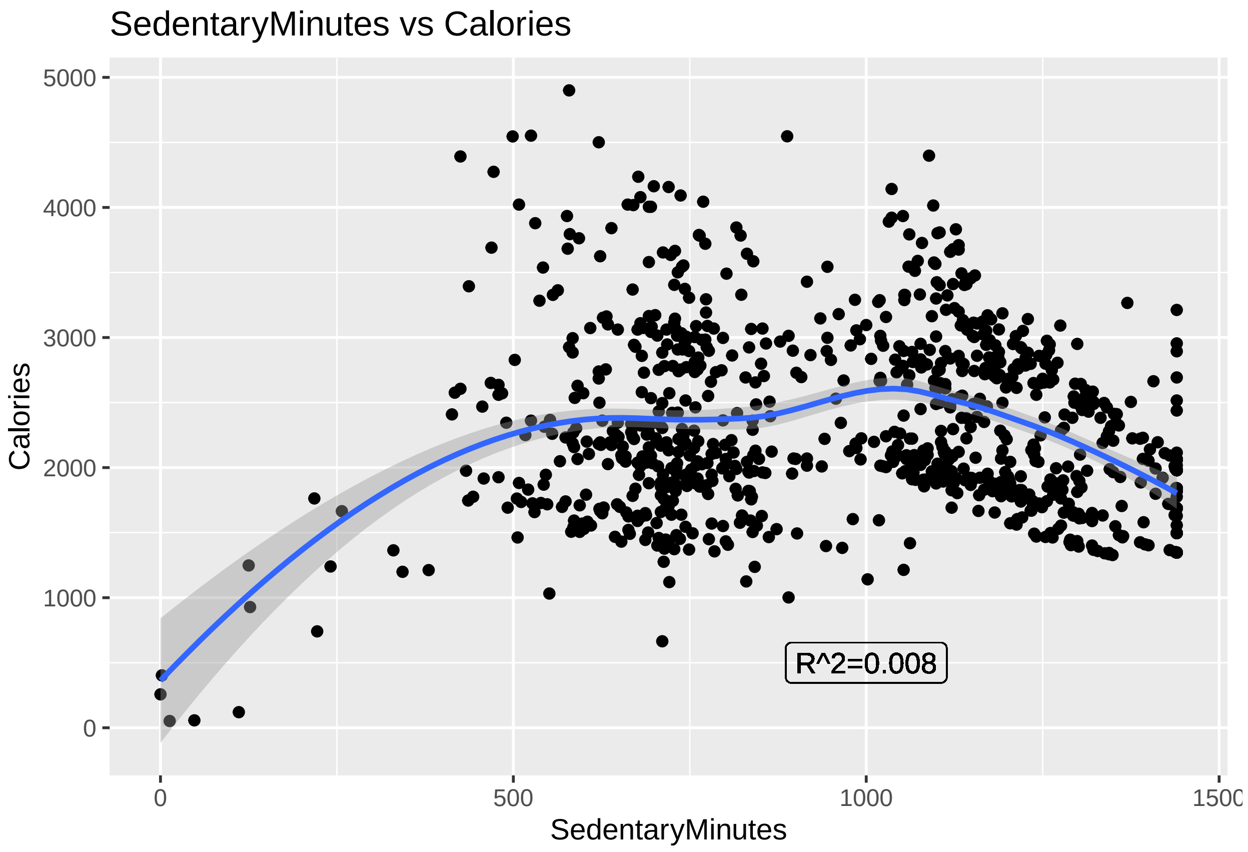
```
##                  SedentaryMinutes    Calories
## SedentaryMinutes       1.00000000 -0.08892396
## Calories              -0.08892396  1.00000000
```

```
rsq(in_s_cal)
```

```
##                   SedentaryMinutes   Calories
## SedentaryMinutes       1.00000000 0.00790747
## Calories               0.00790747 1.00000000
```

```
ggplot(in_s_cal, aes(x = SedentaryMinutes, y = Calories)) + geom_point() +
    geom_smooth() + labs(title = "SedentaryMinutes vs Calories") +
    geom_label(label = "R^2=0.008", x = 1000, y = 500, fill = NA)
```

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



The conclusion is that sedentary minutes do not have a strong correlation with calories and sedentary minutes are a bad predictor of the calorie consumption.

```
in_la_cal <- daily_activity_clean %>%
    select(LightlyActiveMinutes, Calories)

cor(in_la_cal)
```

Lightly Active Minutes vs Calories

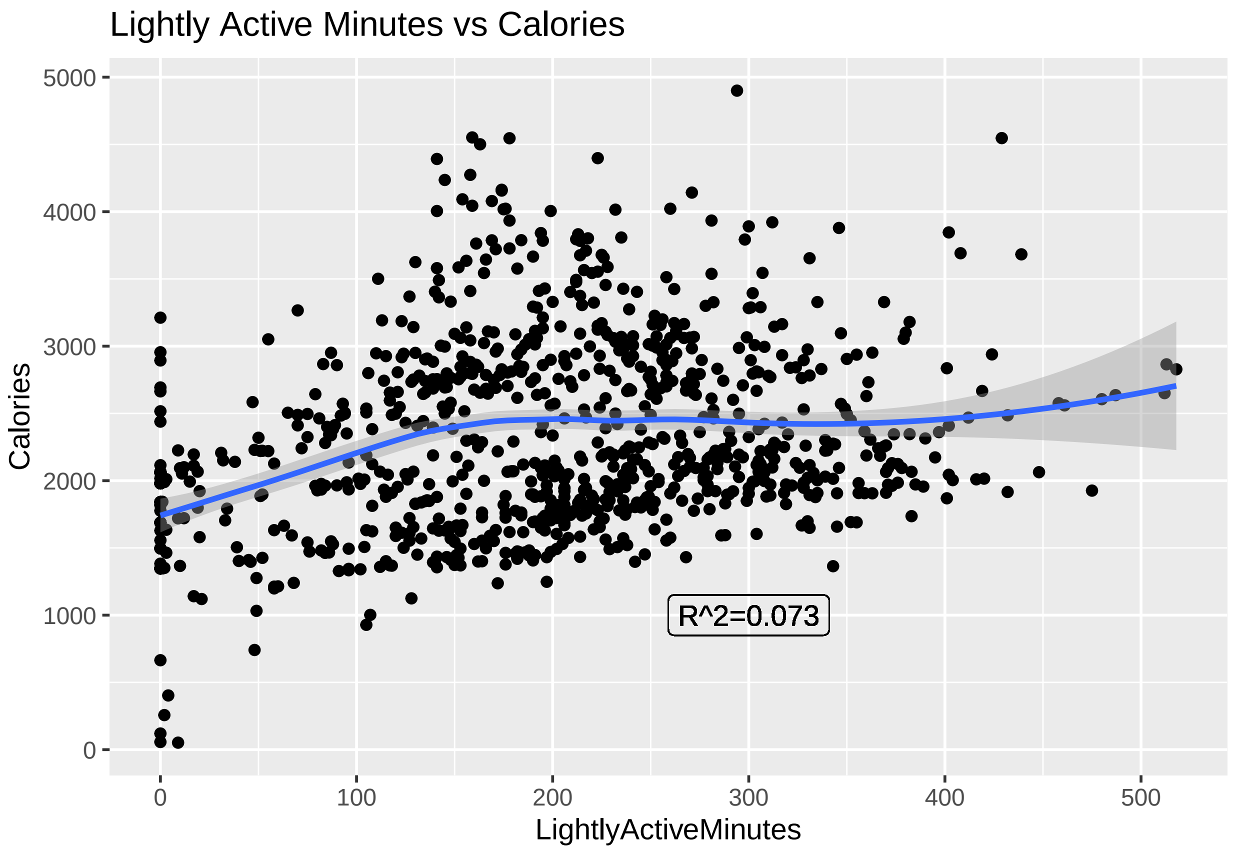
```
##                       LightlyActiveMinutes  Calories
## LightlyActiveMinutes             1.0000000 0.2702665
## Calories                         0.2702665 1.0000000
```

```
rsq(in_la_cal)
```

```
##                        LightlyActiveMinutes   Calories
## LightlyActiveMinutes             1.00000000 0.07304395
## Calories                         0.07304395 1.00000000
```

```
ggplot(in_la_cal, aes(x = LightlyActiveMinutes, y = Calories)) +
    geom_point() + geom_smooth() + labs(title = "Lightly Active Minutes vs Calories") +
    geom_label(label = "R^2=0.073", x = 300, y = 1000, fill = NA)
```

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



It is a stronger predictor than sedentary minutes but does not correlate strongly with calories.

```
in_fa_cal <- daily_activity_clean %>%
    select(FairlyActiveMinutes, Calories)

cor(in_fa_cal)
```

Fairly Active Minutes vs Calories

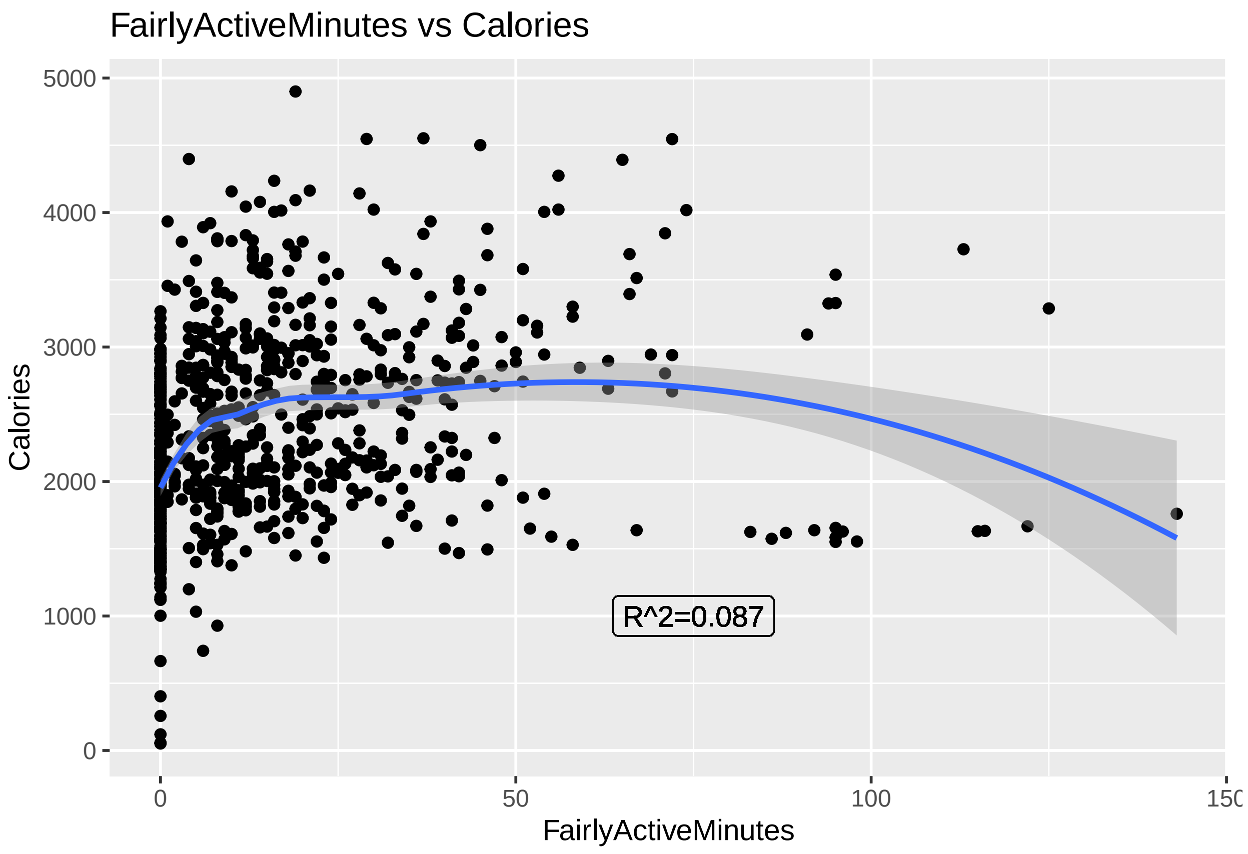
```
##                       FairlyActiveMinutes Calories
## FairlyActiveMinutes              1.000000 0.295164
## Calories                         0.295164 1.000000
```

```
rsq(in_fa_cal)
```

```
##                       FairlyActiveMinutes   Calories
## FairlyActiveMinutes            1.00000000 0.08712181
## Calories                       0.08712181 1.00000000
```

```
ggplot(in_fa_cal, aes(x = FairlyActiveMinutes, y = Calories)) +
    geom_point() + geom_smooth() + labs(title = "FairlyActiveMinutes vs Calories") +
    geom_label(label = "R^2=0.087", x = 75, y = 1000, fill = NA)
```

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



The strongest correlation so far, but once again not a strong predictor.

```
in_va_cal <- daily_activity_clean %>%
    select(VeryActiveMinutes, Calories)

cor(in_va_cal)
```

VeryActiveMinutes vs Calories

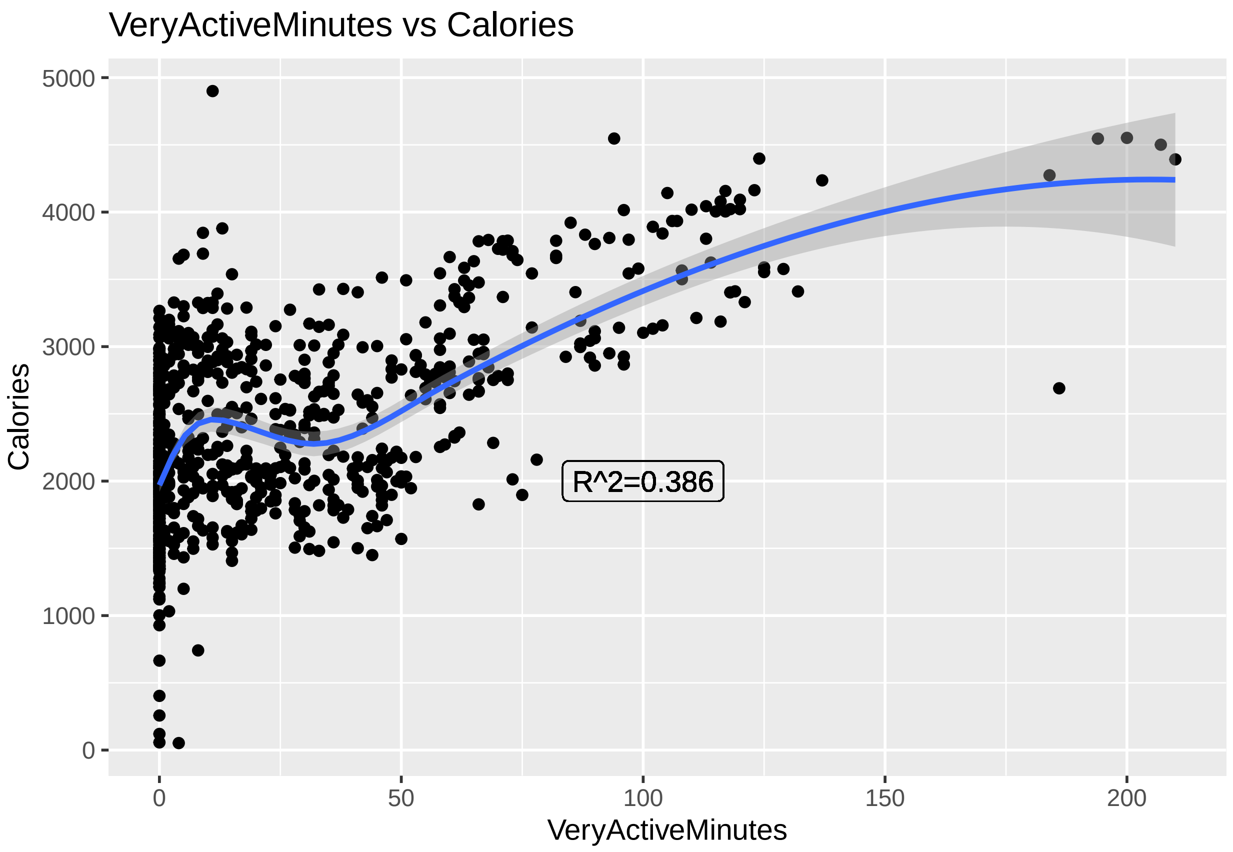
```
##                   VeryActiveMinutes  Calories
## VeryActiveMinutes         1.0000000 0.6213645
## Calories                  0.6213645 1.0000000
```

```
rsq(in_va_cal)
```

```
##                   VeryActiveMinutes  Calories
## VeryActiveMinutes         1.0000000 0.3860939
## Calories                  0.3860939 1.0000000
```

```
ggplot(in_va_cal, aes(x = VeryActiveMinutes, y = Calories)) +
    geom_point() + geom_smooth() + labs(title = "VeryActiveMinutes vs Calories") +
    geom_label(label = "R^2=0.386", x = 100, y = 2000, fill = NA)
```

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



Very active minutes have the strongest correlation and are the best-suited predictor out of all the activity minutes levels.
Sleep has also been analyzed as the dependent variable, but there was no positive nor significant correlation with the variables, nonetheless these results have been included in the appendix.

Key Findings

- Based on the data set we have created a new clustering
  - Kilometers
    - 0-3.499 kilometers=> rarely active
    - 3.5-6.499 kilometers=> lightly active
    - 6.5->=10 kilometers=> active
    - 10->=10 kilometers=> very active
    - 0-4.999 steps=> rarely active
  - Steps
    - 0-4.999 steps=> rarely active
    - 5.000-7.999 steps=> lightly_active
    - 8.000-12.000 steps=> active

        * 12.000->= 12.000 steps=> very active

- People consume the most calories at 12 PM and in the period of 5 PM-7 PM
- People walk the most steps at 12 PM and in the period between 5 PM-7 PM
- The most active minutes during the day are accumulated during 12pm and the time period of 5pm-7pm
- People are the most active on Mondays and Tuesdays, the least on Sundays
- Total distance and calories do have a acceptable relationship
- Total Steps and calories do have a acceptable relationship
- Different levels of activities and calories do have a good relationship
    - The activity level "Very active" has the strongest relationship with calories

Act Phase

The analysis has gained significant insights into how people use their smart devices. The following paragraph will use the findings and outline possible actions for the Bellabeat app.
The Bellabeat app provides users with their health data related to their activity, steps, habits etc. it allows users to understand their current habits and guides them to healthy decisions.
The app allows us to apply our gained knowledge the best. Therefore we make the following suggestions;

- Sending reminder for food consumption
    - Calorie consumption based on activity levels As we already have established, activity levels do have an acceptable correlation with calories. The higher the activity level, the bigger the correlation gets. Based on that knowledge, the app should send a reminder for every bigger meal like breakfast, lunch and dinner. Additional it should send around 11:30 a notification of how many calories have been already consumed the same before 5pm (5pm-7pm the most activity time) and a reminder after the workout at 7pm.
- Calorie consumption based on distance and steps
  The newly created clusters can here be applied; every time a new cluster level has reached, a notification should be sent out for example, if the user has already eaten, as already established total distance and total steps do have a reasonable correlation with calories.
- Giving a weekly overview for users
  It can be helpful to let, every user see their statistic over a week, which will give them a sense of accountability. Based on that, the app can provide suggestions where they need to improve, more steps, more active minutes etc.
  This can be measured first on the analyzed data and eventually will be substituted with primary data, which will be gathered through the app.
- Motivating users to move
  A clustering was already introduced. People should reach every day at least the "active" cluster(8.000-12.000 steps, 6.5-10.000 kilometres). We know the people are very active on Mondays and Tuesdays but do the least on Sundays. Therefore the app should notify people when they seem not to reach the "active" status. This can be done by showing how many steps|kilometres are missing to the goal, and letting them know they have reached their goals on previous days, which is supposed to work as a motivator.

Further/Future Analysis

In the future, we should focus more on our target group, women. The data set, which was analyzed, gave us a great insight about smart device users, but it did not mention how many women participated in the data collection. Moving forward, we should compare our primary data to the data set and look for similarities or different trends.
Collecting primary data/data of our users can be obtained through surveys and analyzing the data gathered through the app.

Appendix

Sleep vs Calories

To analyze the sleep and calory variables the most efficient way, it is the recommended approach to merge both data sets.

```
act_sleep_df <- merge(x = daily_activity_clean, y = daily_sleep_clean,
    c("Id"))
head(act_sleep_df)
```

Merging the datasets & Calculation

```
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
## 1 1503960366    4/29/2016      11181          7.15            7.15
## 2 1503960366    4/29/2016      11181          7.15            7.15
## 3 1503960366    4/29/2016      11181          7.15            7.15
## 4 1503960366    4/29/2016      11181          7.15            7.15
## 5 1503960366    4/29/2016      11181          7.15            7.15
## 6 1503960366    4/29/2016      11181          7.15            7.15
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               1.06                      0.5
## 2                        0               1.06                      0.5
## 3                        0               1.06                      0.5
## 4                        0               1.06                      0.5
## 5                        0               1.06                      0.5
## 6                        0               1.06                      0.5
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                5.58                       0                16
## 2                5.58                       0                16
## 3                5.58                       0                16
## 4                5.58                       0                16
## 5                5.58                       0                16
## 6                5.58                       0                16
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories
## 1                  12                  243              815     1837
## 2                  12                  243              815     1837
## 3                  12                  243              815     1837
## 4                  12                  243              815     1837
## 5                  12                  243              815     1837
## 6                  12                  243              815     1837
##                SleepDay TotalSleepRecords TotalMinutesAsleep TotalTimeInBed
## 1 4/12/2016 12:00:00 AM                 1                327            346
## 2 4/13/2016 12:00:00 AM                 2                384            407
## 3 4/15/2016 12:00:00 AM                 1                412            442
## 4 4/16/2016 12:00:00 AM                 2                 340              367
## 5 4/17/2016 12:00:00 AM                 1                 700              712
## 6 4/19/2016 12:00:00 AM                 1                 304              320
```

```
act_sleep <- act_sleep_df %>%
    select(TotalMinutesAsleep, Calories)

cor(act_sleep)
```

```
##                      TotalMinutesAsleep   Calories
## TotalMinutesAsleep           1.00000000 0.01966779
## Calories                     0.01966779 1.00000000
```

```
rsq(act_sleep)
```

```
##                      TotalMinutesAsleep     Calories
## TotalMinutesAsleep         1.0000000000 0.0003868218
## Calories                   0.0003868218 1.0000000000
```

Sleep as a Dependent Variable

Total Distance vs Sleep

```
td_sleep <- act_sleep_df %>%
    select(TotalDistance, TotalMinutesAsleep)

cor(td_sleep)
```

```
##                      TotalDistance TotalMinutesAsleep
## TotalDistance           1.00000000        -0.09748388
## TotalMinutesAsleep     -0.09748388         1.00000000
```

```
rsq(td_sleep)
```

```
##                      TotalDistance TotalMinutesAsleep
## TotalDistance          1.000000000        0.009503106
## TotalMinutesAsleep     0.009503106        1.000000000
```

Total distance does not have a strong relationship with the sleeping time of the participants; it instead goes the opposite way.

Intensity levels vs Sleep

```
int_s_sleep <- act_sleep_df %>%
    select(SedentaryMinutes, TotalMinutesAsleep)

cor(int_s_sleep)
```

Sedentary Minutes vs Sleep

```
##                      SedentaryMinutes TotalMinutesAsleep
## SedentaryMinutes            1.0000000         -0.1215146
## TotalMinutesAsleep         -0.1215146          1.0000000
```

```
rsq(int_s_sleep)
```

```
##                      SedentaryMinutes TotalMinutesAsleep
## SedentaryMinutes           1.00000000         0.01476579
## TotalMinutesAsleep       0.01476579         1.00000000
```

```
int_la_sleep <- act_sleep_df %>%
    select(LightlyActiveMinutes, TotalMinutesAsleep)

cor(int_la_sleep)
```

Lightly Active Minutes vs Sleep

```
##                        LightlyActiveMinutes TotalMinutesAsleep
## LightlyActiveMinutes           1.00000000         0.02892791
## TotalMinutesAsleep             0.02892791         1.00000000
```

```
rsq(int_la_sleep)
```

```
##                        LightlyActiveMinutes TotalMinutesAsleep
## LightlyActiveMinutes         1.0000000000       0.0008368238
## TotalMinutesAsleep           0.0008368238       1.0000000000
```

```
int_fa_sleep <- act_sleep_df %>%
    select(FairlyActiveMinutes, TotalMinutesAsleep)

cor(int_fa_sleep)
```

Fairly Active Minutes vs Sleep

```
##                       FairlyActiveMinutes TotalMinutesAsleep
## FairlyActiveMinutes           1.0000000         -0.1796766
## TotalMinutesAsleep           -0.1796766          1.0000000
```

```
rsq(int_fa_sleep)
```

```
##                       FairlyActiveMinutes TotalMinutesAsleep
## FairlyActiveMinutes          1.00000000         0.03228367
## TotalMinutesAsleep           0.03228367         1.00000000
```

```
int_va_sleep <- act_sleep_df %>%
    select(VeryActiveMinutes, TotalMinutesAsleep)

cor(int_va_sleep)
```

Very Active Minutes vs Sleep

```
##                      VeryActiveMinutes TotalMinutesAsleep
## VeryActiveMinutes           1.00000000        -0.02571567
## TotalMinutesAsleep         -0.02571567         1.00000000
```

```
rsq(int_va_sleep)
```

```
##                      VeryActiveMinutes TotalMinutesAsleep
## VeryActiveMinutes         1.0000000000       0.0006612956
## TotalMinutesAsleep        0.0006612956       1.0000000000
```

```r
ts_sleep <- act_sleep_df %>%
    select(TotalSteps, TotalMinutesAsleep)

cor(ts_sleep)
```

Total Steps

```
##                     TotalSteps TotalMinutesAsleep
## TotalSteps           1.0000000         -0.1007889
## TotalMinutesAsleep -0.1007889          1.0000000
```

```r
rsq(ts_sleep)
```

```
##                     TotalSteps TotalMinutesAsleep
## TotalSteps         1.00000000         0.01015839
## TotalMinutesAsleep 0.01015839         1.00000000
```

Sources

- Mukaka, M.M. (2012) Statistics corner: A guide to appropriate use of correlation coefficient in medical research. Malawi medical journal : the journal of Medical Association of Malawi, [online] 24(3), pp.69–71. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3576830/
- Fernando, J. (2021) R-Squared Definition. [online] Investopedia. Available at: https://www.investopedia.com/terms/r/r-squared.asp.
- Furberg, R., Brinton, J., Keating, M. and Ortiz, A. (2016). Crowd-sourced Fitbit datasets 03.12.2016-05.12.2016. [online] Zenodo. Available at: https://zenodo.org/record/53894#.X9oeh3Uzaao.