Bellabeat

Pilar Wilches

2023-10-03

Analysis Process for the case study: Bellabeat

Goal: Analyse data from smart devices to understand consumer use of smart device

First of all we install packages and libraries for data clean and visualization

```
options(repos = "https://cran.rstudio.com/")

install.packages("here")
```

```
##
## The downloaded binary packages are in
##  /var/folders/fs/hqldbrgx41l3g65ry__w83l40000gn/T//Rtmpcjhs1V/downloaded_packages
```

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

```
##
## The downloaded binary packages are in
##  /var/folders/fs/hqldbrgx41l3g65ry__w83l40000gn/T//Rtmpcjhs1V/downloaded_packages
```

```
install.packages("janitor")
```

```
##
## The downloaded binary packages are in
##  /var/folders/fs/hqldbrgx41l3g65ry__w83l40000gn/T//Rtmpcjhs1V/downloaded_packages
```

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

```
##
## The downloaded binary packages are in
##  /var/folders/fs/hqldbrgx41l3g65ry__w83l40000gn/T//Rtmpcjhs1V/downloaded_packages
```

```
library("here")
```

```
## here() starts at /Users/maria/Bellabeat
```

```
library("skimr")
library("janitor")
```

```
##
## Attaching package: 'janitor'
```

```
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library("dplyr")
```

```
##
```

```
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library("lubridate")
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library("readr")
library("ggplot2")
```

Work with file dailyIntensities_merged.csv

1. Import Data and previsualization

Now, I import the file: dailyIntensities_merged.csv and create the data frame, With this data frame I want know the relationship between the days of the week and the person activity.

```
setwd("/Users/maria/Bellabeat")
dailyIntensities_df <- read_csv("dailyIntensities_merged.csv")
```

```
## Rows: 940 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ActivityDay
## dbl (9): Id, SedentaryMinutes, LightlyActiveMinutes, FairlyActiveMinutes, Ve...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Then, I preview the data with different functions for review cols, rows, type of data.

```
View(dailyIntensities_df)

glimpse(dailyIntensities_df)
```

```
## Rows: 940
## Columns: 10
## $ Id                       <dbl> 1503960366, 1503960366, 1503960366, 150396036~
## $ ActivityDay              <chr> "4/12/2016", "4/13/2016", "4/14/2016", "4/15/~
## $ SedentaryMinutes         <dbl> 728, 776, 1218, 726, 773, 539, 1149, 775, 818~
## $ LightlyActiveMinutes     <dbl> 328, 217, 181, 209, 221, 164, 233, 264, 205, ~
## $ FairlyActiveMinutes      <dbl> 13, 19, 11, 34, 10, 20, 16, 31, 12, 8, 27, 21~
## $ VeryActiveMinutes        <dbl> 25, 21, 30, 29, 36, 38, 42, 50, 28, 19, 66, 4~
## $ SedentaryActiveDistance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ LightActiveDistance      <dbl> 6.06, 4.71, 3.91, 2.83, 5.04, 2.51, 4.71, 5.0~
## $ ModeratelyActiveDistance <dbl> 0.55, 0.69, 0.40, 1.26, 0.41, 0.78, 0.64, 1.3~
## $ VeryActiveDistance       <dbl> 1.88, 1.57, 2.44, 2.14, 2.71, 3.19, 3.25, 3.5~
```

```r
colnames(dailyIntensities_df)
```

```
##  [1] "Id"                       "ActivityDay"             
##  [3] "SedentaryMinutes"         "LightlyActiveMinutes"    
##  [5] "FairlyActiveMinutes"      "VeryActiveMinutes"       
##  [7] "SedentaryActiveDistance"  "LightActiveDistance"     
##  [9] "ModeratelyActiveDistance" "VeryActiveDistance"
```

2. Data cleaning

I check the consistence of data frame

```r
clean_names(dailyIntensities_df)
```

```
## # A tibble: 940 x 10
##            id activity_day sedentary_minutes lightly_active_minutes
##         <dbl> <chr>                    <dbl>                  <dbl>
##  1 1503960366 4/12/2016                  728                    328
##  2 1503960366 4/13/2016                  776                    217
##  3 1503960366 4/14/2016                 1218                    181
##  4 1503960366 4/15/2016                  726                    209
##  5 1503960366 4/16/2016                  773                    221
##  6 1503960366 4/17/2016                  539                    164
##  7 1503960366 4/18/2016                 1149                    233
##  8 1503960366 4/19/2016                  775                    264
##  9 1503960366 4/20/2016                  818                    205
## 10 1503960366 4/21/2016                  838                    211
## # i 930 more rows
## # i 6 more variables: fairly_active_minutes <dbl>, very_active_minutes <dbl>,
## #   sedentary_active_distance <dbl>, light_active_distance <dbl>,
## #   moderately_active_distance <dbl>, very_active_distance <dbl>
```

When I apply this function, I see that the name of the columns can be improved according to the syntax of clean code, so I change the column names type snake case

```r
 rename_columns_to_snake_case <- function(dataframe){
  dataframe %>%
    rename_with(~tolower(gsub("([a-z0-9])([A-Z])", "\\1_\\2", .)),.cols = everything())
 }

dailyIntensities_new_df <- rename_columns_to_snake_case(dailyIntensities_df)

View(dailyIntensities_new_df)
```

3. Manipulation/Processing of data

As I want see the day of week has more or less activity then I work with the field activity_data and I convert this field char in date. For this I use the library lubridate.

```r
dailyIntensities_new_df <- dailyIntensities_new_df %>%
  mutate(day_of_week = weekdays(as.Date(activity_day, format="%m/%d/%Y")) )
```

Now, it is very important to have statistics, below is the comparison:

Day of week vrs. Average active minutes

```
mean_active_minutes_df <- dailyIntensities_new_df %>%
  group_by(day_of_week) %>%
  summarize(mean_active_minutes =      round(mean(very_active_minutes+fairly_active_minutes+lightly_acti
  arrange(mean_active_minutes)
```

Day of week vrs. Sedentary minutes

```
dailyIntensities_new_df %>%
  group_by(day_of_week) %>%
  summarize(mean_sedentary_minutes =round(mean(sedentary_minutes))) %>%
  arrange(desc(mean_sedentary_minutes))
```

```
## # A tibble: 7 x 2
##   day_of_week mean_sedentary_minutes
##   <chr>                        <dbl>
## 1 Monday                        1028
## 2 Tuesday                       1007
## 3 Friday                        1000
## 4 Sunday                         990
## 5 Wednesday                      989
## 6 Saturday                       964
## 7 Thursday                       962
```

Work with file dailyActivity_merged.csv

1. Import Data and previsualization

Now, I import the file: dailyActivity_merged.csv and create the data frame, With this data frame I want know the relationship between the days of the week and the calories, steps and activity in general.

```
setwd("/Users/maria/Bellabeat")
dailyActivity_df <- read_csv("dailyActivity_merged.csv")
```

```
## Rows: 940 Columns: 15
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): ActivityDate
## dbl (14): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesDi...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Then, I preview the data with different functions for review cols, rows, type of data.

```
View(dailyActivity_df)

glimpse(dailyActivity_df)
```

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

This visualization is very important, for instance, the column ActivityDay is character and must be date type for the analysis.

```
colnames(dailyActivity_df)
```

```
##  [1] "Id"                       "ActivityDate"
##  [3] "TotalSteps"               "TotalDistance"
##  [5] "TrackerDistance"          "LoggedActivitiesDistance"
##  [7] "VeryActiveDistance"       "ModeratelyActiveDistance"
##  [9] "LightActiveDistance"      "SedentaryActiveDistance"
## [11] "VeryActiveMinutes"        "FairlyActiveMinutes"
## [13] "LightlyActiveMinutes"     "SedentaryMinutes"
## [15] "Calories"
```

2. Data cleaning

I check the consistence of data frame

```
clean_names(dailyActivity_df)
```

```
## # A tibble: 940 x 15
##            id activity_date total_steps total_distance tracker_distance
##         <dbl> <chr>               <dbl>          <dbl>            <dbl>
##  1 1503960366 4/12/2016           13162           8.5              8.5
##  2 1503960366 4/13/2016           10735           6.97             6.97
##  3 1503960366 4/14/2016           10460           6.74             6.74
##  4 1503960366 4/15/2016            9762           6.28             6.28
##  5 1503960366 4/16/2016           12669           8.16             8.16
##  6 1503960366 4/17/2016            9705           6.48             6.48
##  7 1503960366 4/18/2016           13019           8.59             8.59
##  8 1503960366 4/19/2016           15506           9.88             9.88
##  9 1503960366 4/20/2016           10544           6.68             6.68
## 10 1503960366 4/21/2016            9819           6.34             6.34
## # i 930 more rows
## # i 10 more variables: logged_activities_distance <dbl>,
## #   very_active_distance <dbl>, moderately_active_distance <dbl>,
## #   light_active_distance <dbl>, sedentary_active_distance <dbl>,
## #   very_active_minutes <dbl>, fairly_active_minutes <dbl>,
## #   lightly_active_minutes <dbl>, sedentary_minutes <dbl>, calories <dbl>
```

When I apply this function, I see that the name of the columns can be improved according the syntax of clean code, so I apply the function created previously, which changes the column names like snake case

```
dailyActivity_new_df <- rename_columns_to_snake_case(dailyActivity_df)

View(dailyActivity_new_df)
```

3. Manipulation/Processing of data

And this point, I used the same code for change and add the column name day_of_week

```
dailyActivity_new_df <- dailyActivity_new_df %>%
  mutate(day_of_week = weekdays(as.Date(activity_date, format="%m/%d/%Y")) )
```

Check the new column:

```
 View(dailyActivity_new_df)
```

Now, it's statistics' time:

Day of week vrs. Calories

```
 dailyActivity_new_df %>%
  group_by(day_of_week) %>%
  summarize(mean_calories = round(mean(calories,na.rm = TRUE)),
            mean_steps = round(mean(total_steps,na.rm = TRUE))) %>%
  arrange(desc(mean_calories))
```

```
## # A tibble: 7 x 3
##   day_of_week mean_calories mean_steps
##   <chr>               <dbl>      <dbl>
## 1 Tuesday              2356       8125
## 2 Saturday             2355       8153
## 3 Friday               2332       7448
## 4 Monday               2324       7781
## 5 Wednesday            2303       7559
## 6 Sunday               2263       6933
## 7 Thursday             2200       7406
```

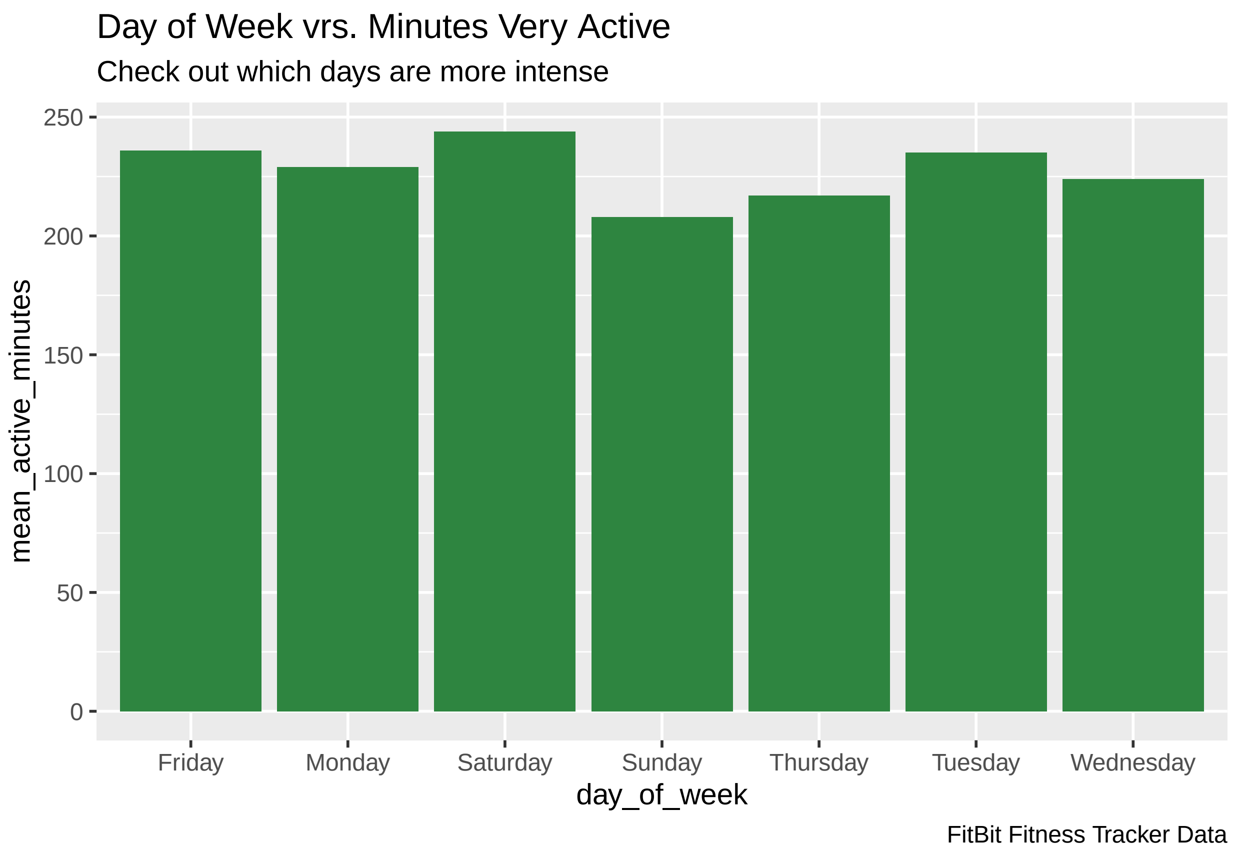
```
 mean_calories_df <- dailyActivity_new_df %>%
  group_by(day_of_week) %>%
  summarize(mean_calories = round(mean(calories, na.rm = TRUE))) %>%
  arrange(desc(mean_calories))

  mean_total_steps_df <- dailyActivity_new_df %>%
  group_by(day_of_week) %>%
  summarize(mean_steps = round(mean(total_steps, na.rm = TRUE))) %>%
  arrange(desc(mean_steps))
```

Visualization of data

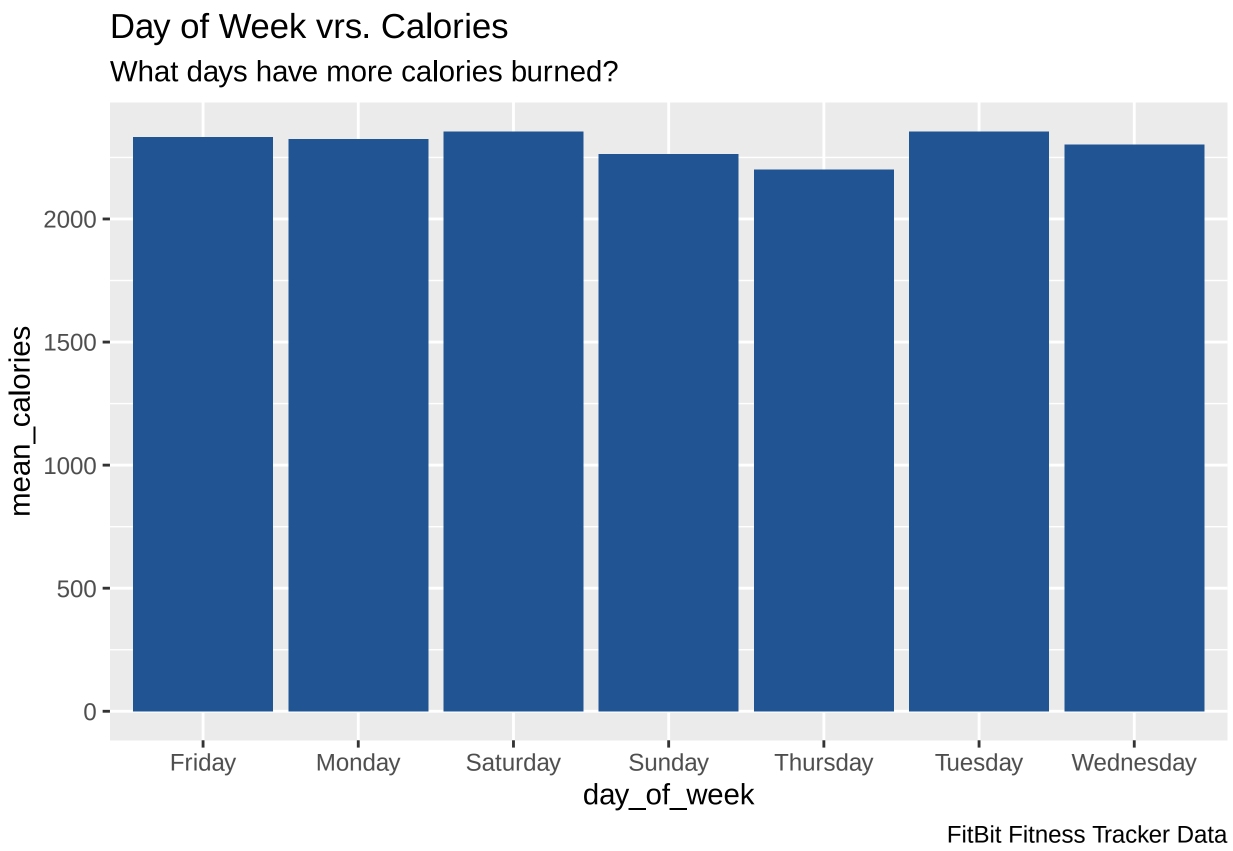
In this step, I'll show the different graphics of my data:

ggplot

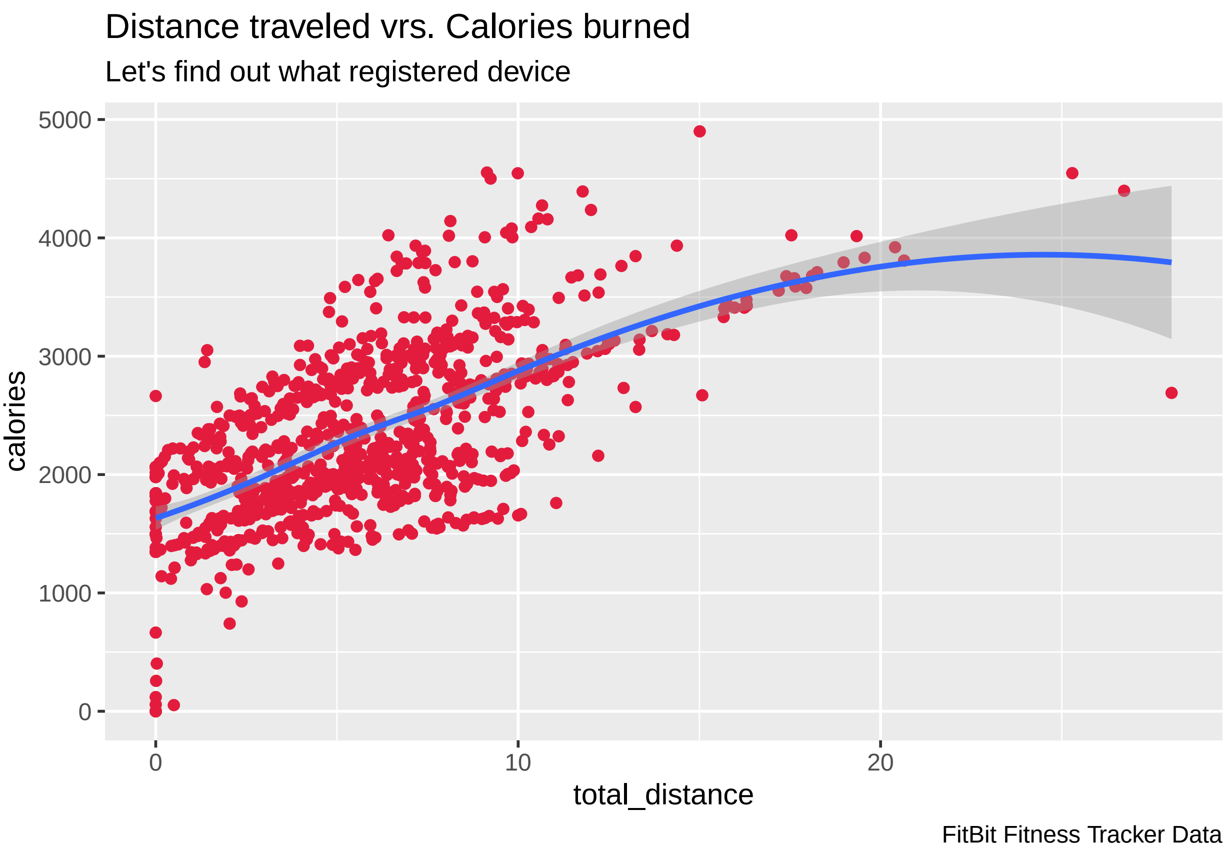
```
ggplot(data = mean_active_minutes_df) +
  geom_col(mapping = aes(x= day_of_week, y=mean_active_minutes), fill="#2e8540")+
  labs(title = "Day of Week vrs. Minutes Very Active",
       subtitle = "Check out which days are more intense",
       caption = "FitBit Fitness Tracker Data")
```



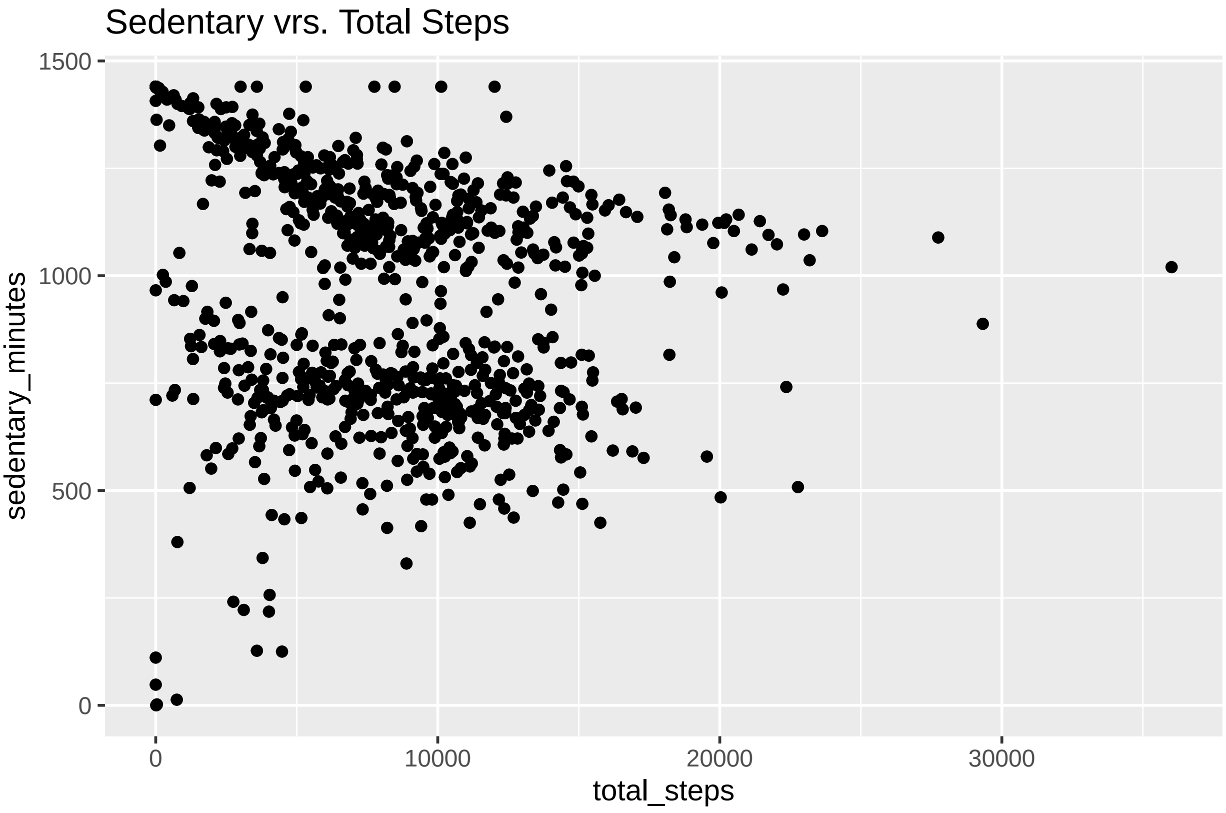
```r
ggplot(data = mean_calories_df) +
  geom_col(mapping = aes(x= day_of_week, y=mean_calories), fill="#205493")+
  labs(title = "Day of Week vrs. Calories",
       subtitle = "What days have more calories burned?",
       caption = "FitBit Fitness Tracker Data")
```



```
ggplot(data = dailyActivity_new_df) +
  geom_point(mapping = aes(x= total_distance, y=calories), color="#e31c3d") +
  geom_smooth(mapping = aes(x= total_distance, y=calories)) +
  labs(title = "Distance traveled vrs. Calories burned",
       subtitle = "Let's find out what registered device",
       caption = "FitBit Fitness Tracker Data")
```



```r
ggplot(data=dailyActivity_new_df, aes(x=total_steps, y=sedentary_minutes)) +
  geom_point() +
  labs(title = "Sedentary vrs. Total Steps ",
       caption = "FitBit Fitness Tracker Data")
```



FitBit Fitness Tracker Data

Technical Notes

- I could have named all variables with clean code I was able to used a function to change the data to the day of week and not repeat code.
- As I progressed in the project, I became more familiar with the files, so I realized note that I could have started with the activity file and not Intensitive file csv.
- In data cleaning I didn't delete the day with zero activity because I don't know if it is a mistake, this situation in a real case, I would ask the organization.

Conclusions

- Develop sporting events on Saturdays. This day presents the highest intensity of movement reported in the smart devices. The figures reveal that on average on Saturday, women have 240 minutes of high physical activity (4 hours)
- Thursdays are the day on average that the fewest calories are burned. But how do users know if they are burning enough calories to know if they have healthy habits? The strategy I propose is that on Thursdays, through email and the company's social media accounts, to show content on healthy habits in accordance with what is suggested by the OMS.
- Create an awareness campaign, making visible the days of greatest sedentary lifestyle.