# Bellabeat Case Study

Sean Perry

2025-08-04

## Task Summary

Task: Analyze smart device data to figure out how people are using their smart devices Goal: Find insights to help guide marketing strategy

Questions: 1. What are some trends in smart device usage? 2. How could these trends apply to Bellabeat customers? 3. How could these trends help influence Bellabeat marketing strategy?

## ASK

What is the problem you're trying to solve? - I'm trying to analyze the FitBit data submitted by 30 people to find trends in how they are using their smart devices and apply that insight to Bellabeat and come up with potential ways they can adjust their marketing strategy to accommodate the way people are already using their smart devices.

How can your insights drive business decisions? - Ideally, the trends and insights that I find from analyzing the FitBit data will allow me to make informed suggestions that Bellabeat can use to adjust their marketing strategy and ultimately reach more people/sell more devices.

## PREPARE

Where is your data stored? It's stored primarily in Kaggle, secondarily on my computer, and in posit/Rstudio for analysis.

How is the data organized? Is it in long or wide format? It's stored in 11 separate csv files based on the subject of the data and all files are in long format.

Are there issues with bias or credibility in this data? Does your data ROCCC? Considering the data is from 30 people who agreed to have their data be used, there is definitely a bias - people who are likely to be okay with the data from their smart device being used are probably the kind of people who actively use their smart device and find it beneficial, so it may not be truly representative of all people who have smart devices. It's only 30 people, which is a small sample size. We also don't know anything about these people besides the information from their smart device - no demographic information such as age, race, gender, etc. - so the conclusions we can draw from it are somewhat limited by that. The data doesn't fully ROCCC - it's from 2016 and a lot has happened between 2016 and now that could change the way people use smart devices/are being active, especially in 2020 and 2021.

How are you addressing licensing, privacy, security, and accessibility? The data is publicly available and doesn't have any identifying information, so I don't think it's terribly relevant here.

How did you verify the data's integrity? I looked at each dataset using the head(), colnames(), duplicated(), sum(is.na()), and summary() to get a general feel for what's included and how much, if any, data is missing. Summary() in particular is helpful to see the general summary statistics, which helps to find outliers/anomalous records. I also checked that the values I see in the summary align with what I would generally expect for fitness data for standard users.

How does it help you answer your question? The data gives us enough to look at the general trends of the users over the course of a month and shows us that users with a variety of activity levels can all benefit from the device.

Are there any problems with the data? Definitely. Despite how large some of the datasets are, the actual number of people whose data is included is pretty small - only 30 people. As mentioned before, there's also no demographic information, which would help in the analysis and subsequent recommendations to be able to target promotions/material more specifically.

### Installing and loading necessary packages

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.2     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggplot2)
library(skimr)
```

### Import data for 3-12 to 4-11

I'm not using the minute datasets or heartrate_seconds because they're too big for Rstudio (especially heartrate_seconds) and too granular to provide meaningful insights that the hourly and daily cannot.

```
daily_activity <- read_csv("Bellabeat Data/3-12 to 4-11/dailyActivity_merged.csv")
```

```
## Rows: 457 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.
```

```
hourly_calories <- read_csv("Bellabeat Data/3-12 to 4-11/hourlyCalories_merged.csv")
```

```
## Rows: 24084 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ActivityHour
## dbl (2): Id, Calories
##
## 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.
```

```
hourly_intensities <- read_csv("Bellabeat Data/3-12 to 4-11/hourlyIntensities_merged.csv")
```

```
## Rows: 24084 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ActivityHour
## dbl (3): Id, TotalIntensity, AverageIntensity
##
## 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.
```

```
hourly_steps <- read_csv("Bellabeat Data/3-12 to 4-11/hourlySteps_merged.csv")
```

```
## Rows: 24084 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ActivityHour
## dbl (2): Id, StepTotal
##
## 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.
```

```
weightlog_info <- read_csv("Bellabeat Data/3-12 to 4-11/weightLogInfo_merged.csv")
```

```
## Rows: 33 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Date
## dbl (6): Id, WeightKg, WeightPounds, Fat, BMI, LogId
## lgl (1): IsManualReport
##
## 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.
```

### Taking a peek at the imported data

```
head(daily_activity)
```

```
## # A tibble: 6 x 15
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
##        <dbl> <chr>             <dbl>         <dbl>           <dbl>
## 1 1503960366 3/25/2016         11004          7.11            7.11
## 2 1503960366 3/26/2016         17609         11.6            11.6
## 3 1503960366 3/27/2016         12736          8.53            8.53
## 4 1503960366 3/28/2016         13231          8.93            8.93
```

```
## 5 1503960366 3/29/2016         12041          7.85            7.85
## 6 1503960366 3/30/2016         10970          7.16            7.16
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

```
head(hourly_calories)
```

```
## # A tibble: 6 x 3
##           Id ActivityHour          Calories
##        <dbl> <chr>                    <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM       48
## 2 1503960366 3/12/2016 1:00:00 AM        48
## 3 1503960366 3/12/2016 2:00:00 AM        48
## 4 1503960366 3/12/2016 3:00:00 AM        48
## 5 1503960366 3/12/2016 4:00:00 AM        48
## 6 1503960366 3/12/2016 5:00:00 AM        48
```

```
head(hourly_intensities)
```

```
## # A tibble: 6 x 4
##           Id ActivityHour          TotalIntensity AverageIntensity
##        <dbl> <chr>                          <dbl>             <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM              0                 0
## 2 1503960366 3/12/2016 1:00:00 AM               0                 0
## 3 1503960366 3/12/2016 2:00:00 AM               0                 0
## 4 1503960366 3/12/2016 3:00:00 AM               0                 0
## 5 1503960366 3/12/2016 4:00:00 AM               0                 0
## 6 1503960366 3/12/2016 5:00:00 AM               0                 0
```

```
head(hourly_steps)
```

```
## # A tibble: 6 x 3
##           Id ActivityHour          StepTotal
##        <dbl> <chr>                     <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM         0
## 2 1503960366 3/12/2016 1:00:00 AM          0
## 3 1503960366 3/12/2016 2:00:00 AM          0
## 4 1503960366 3/12/2016 3:00:00 AM          0
## 5 1503960366 3/12/2016 4:00:00 AM          0
## 6 1503960366 3/12/2016 5:00:00 AM          0
```

```
head(weightlog_info)
```

```
## # A tibble: 6 x 8
##           Id Date       WeightKg WeightPounds   Fat   BMI IsManualReport   LogId
##        <dbl> <chr>         <dbl>        <dbl> <dbl> <dbl> <lgl>            <dbl>
## 1 1503960366 4/5/2016 ~     53.3         118.    22  23.0 TRUE           1.46e12
## 2 1927972279 4/10/2016~    130.          286.    NA  46.2 FALSE          1.46e12
## 3 2347167796 4/3/2016 ~     63.4         140.    10  24.8 TRUE           1.46e12
## 4 2873212765 4/6/2016 ~     56.7         125.    NA  21.5 TRUE           1.46e12
## 5 2873212765 4/7/2016 ~     57.2         126.    NA  21.6 TRUE           1.46e12
## 6 2891001357 4/5/2016 ~     88.4         195.    NA  25.0 TRUE           1.46e12
```

**Initial Observations**

Primary/foreign key for all datasets is ID, as you would expect.

Since all of this data deals with repeated measurements over minutes/hours/days, how is the date/time of the measurement recorded? daily_activity - "ActivityDate" - date hourly_calories - "ActivityHour" - datetime hourly_intensities - "ActivityHour" - datetime hourly_steps - "ActivityHour" - datetime weightlog_info - "Date" - datetime So we'll likely want to split the datetime columns into date and time for consistency and easy of joining later.

General dataset summaries (including all datasets, not just the ones used for analysis, for future reference): - daily_activity shows the total calories used per day, the total number of steps taken and distance walked (sum of logged activities and tracker), and a breakdown of the activity level over that distance as well as the time spent at each activity level in minutes. Does not include time sleeping. - heartrate_seconds is the heart rate at 5-second intervals - hourly_calories shows the amount of calories used every hour - hourly_intensities shows the total and average intensities of activities done during that hour (what are the units of intensity? how is it calculated/determined?) - hourly_steps shows the total number of steps taken during that hour - minute_calories shows the amount of calories used every minute - minute_intensities shows the level of intensity of activity taking place that minute - minute_mets shows the METS (metabolic equivalent of task - the metabolic activity relative to resting) of the activity taking place that minute - minute_sleep shows the level/phase of sleep (I'm guessing - need to confirm) during that minute (taken at the middle of each minute instead of the start, which is interesting) - minute_steps shows the total number of steps taken during that minute - weightlog_info shows the weight in kg and lbs, some measure of fat (for some records), a calculated BMI, and whether or not the data was reported manually for a specific day

Checking the column names

```
colnames(daily_activity)
```

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

```
colnames(hourly_calories)
```

```
## [1] "Id"           "ActivityHour" "Calories"
```

```
colnames(hourly_intensities)
```

```
## [1] "Id"               "ActivityHour"     "TotalIntensity"   "AverageIntensity"
```

```
colnames(hourly_steps)
```

```
## [1] "Id"           "ActivityHour" "StepTotal"
```

```
colnames(weightlog_info)
```

```
## [1] "Id"             "Date"           "WeightKg"       "WeightPounds"
## [5] "Fat"            "BMI"            "IsManualReport" "LogId"
```

Nothing too crazy about the column names.

**First thoughts:**

1. When during the day are people using smart devices most active? Check highest calories, intensities, steps
2. How much are people walking, when, and at what intensities? Compare highest steps with intensity and time

3a. How many weightlog reports are manual vs automatic? 3b. Do the reports for the same people show any interesting trends? 3c. Does any weight/fat loss or gain coincide with higher or lower levels of activity/intensity? Compare changes in weightlog with corresponding intensities/steps/calories

4. Does it seem like people are wearing their smart devices all day or only when doing activities? Check activity/sedentary durations in daily_activity

**Data Verification** Let's count the number of unique IDs, nulls/missing values, and duplicate records in each dataset and check the summaries.

```
print("Daily Activity: ")
```

```
## [1] "Daily Activity: "
```

```
length(table(daily_activity$Id))
```

```
## [1] 35
```

```
table(daily_activity$Id)
```

```
##
## 1503960366 1624580081 1644430081 1844505072 1927972279 2022484408 2026352035
##         19         19         10         12         12         12         12
## 2320127002 2347167796 2873212765 2891001357 3372868164 3977333714 4020332650
##         12         15         12          8         10         12         32
## 4057192912 4319703577 4388161847 4445114986 4558609924 4702921684 5553957443
##         32         12          8         15         12         15         12
## 5577150313 6117666160 6290855005 6391747486 6775888955 6962181067 7007744171
##         11         10         10          9          9         14         12
## 7086361926 8053475328 8253242879 8378563200 8583815059 8792009665 8877689391
##         12         11         12         12          8         12         12
```

```
sum(is.na(daily_activity))
```

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

```
sum(duplicated(daily_activity))
```

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

```
summary(daily_activity)
```

```
##        Id            ActivityDate         TotalSteps    TotalDistance
##  Min.   :1.504e+09   Length:457         Min.   :    0   Min.   : 0.000
##  1st Qu.:2.347e+09   Class :character   1st Qu.: 1988   1st Qu.: 1.410
##  Median :4.057e+09   Mode  :character   Median : 5986   Median : 4.090
##  Mean   :4.629e+09                      Mean   : 6547   Mean   : 4.664
##  3rd Qu.:6.392e+09                      3rd Qu.:10198   3rd Qu.: 7.160
##  Max.   :8.878e+09                      Max.   :28497   Max.   :27.530
##  TrackerDistance LoggedActivitiesDistance VeryActiveDistance
##  Min.   : 0.00   Min.   :0.0000           Min.   : 0.000
##  1st Qu.: 1.28   1st Qu.:0.0000           1st Qu.: 0.000
##  Median : 4.09   Median :0.0000           Median : 0.000
##  Mean   : 4.61   Mean   :0.1794           Mean   : 1.181
```

```
##  3rd Qu.: 7.11   3rd Qu.:0.0000           3rd Qu.: 1.310
##  Max.   :27.53   Max.   :6.7271           Max.   :21.920
##  ModeratelyActiveDistance LightActiveDistance SedentaryActiveDistance
##  Min.   :0.0000           Min.   : 0.00       Min.   :0.000000
##  1st Qu.:0.0000           1st Qu.: 0.87       1st Qu.:0.000000
##  Median :0.0200           Median : 2.93       Median :0.000000
##  Mean   :0.4786           Mean   : 2.89       Mean   :0.001904
##  3rd Qu.:0.6700           3rd Qu.: 4.46       3rd Qu.:0.000000
##  Max.   :6.4000           Max.   :12.51       Max.   :0.100000
##  VeryActiveMinutes FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes
##  Min.   :  0.00    Min.   :  0.00      Min.   :  0.0        Min.   :  32.0
##  1st Qu.:  0.00    1st Qu.:  0.00      1st Qu.: 64.0        1st Qu.: 728.0
##  Median :  0.00    Median :  1.00      Median :181.0        Median :1057.0
##  Mean   : 16.62    Mean   : 13.07      Mean   :170.1        Mean   : 995.3
##  3rd Qu.: 25.00    3rd Qu.: 16.00      3rd Qu.:257.0        3rd Qu.:1285.0
##  Max.   :202.00    Max.   :660.00      Max.   :720.0        Max.   :1440.0
##     Calories
##  Min.   :   0
##  1st Qu.:1776
##  Median :2062
##  Mean   :2189
##  3rd Qu.:2667
##  Max.   :4562
```

```
print("Hourly Calories: ")
```

```
## [1] "Hourly Calories: "
```

```
length(table(hourly_calories$Id))
```

```
## [1] 34
```

```
table(hourly_calories$Id)
```

```
##
## 1503960366 1624580081 1644430081 1844505072 1927972279 2022484408 2026352035
##        744        755        700        751        755        755        754
## 2320127002 2347167796 2873212765 2891001357 3372868164 3977333714 4020332650
##        755        751        740         12        711        747        749
## 4057192912 4319703577 4445114986 4558609924 4702921684 5553957443 5577150313
##        750        671        754        754        742        752        730
## 6117666160 6290855005 6391747486 6775888955 6962181067 7007744171 7086361926
##        695        521        675        682        754        752        752
## 8053475328 8253242879 8378563200 8583815059 8792009665 8877689391
##        754        740        754        666        754        753
```

```
sum(is.na(hourly_calories))
```

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

```
sum(duplicated(hourly_calories))
```

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

```
summary(hourly_calories)
```

```
##        Id            ActivityHour          Calories
##  Min.   :1.504e+09   Length:24084       Min.   : 42.00
```

```
##  1st Qu.:2.347e+09   Class :character   1st Qu.: 61.00
##  Median :4.559e+09   Mode  :character   Median : 77.00
##  Mean   :4.889e+09                      Mean   : 94.27
##  3rd Qu.:6.962e+09                      3rd Qu.:104.00
##  Max.   :8.878e+09                      Max.   :933.00
```

```
print("Hourly Intensities: ")
```

```
## [1] "Hourly Intensities: "
```

```
length(table(hourly_intensities$Id))
```

```
## [1] 34
```

```
table(hourly_intensities$Id)
```

```
##
## 1503960366 1624580081 1644430081 1844505072 1927972279 2022484408 2026352035
##        744        755        700        751        755        755        754
## 2320127002 2347167796 2873212765 2891001357 3372868164 3977333714 4020332650
##        755        751        740         12        711        747        749
## 4057192912 4319703577 4445114986 4558609924 4702921684 5553957443 5577150313
##        750        671        754        754        742        752        730
## 6117666160 6290855005 6391747486 6775888955 6962181067 7007744171 7086361926
##        695        521        675        682        754        752        752
## 8053475328 8253242879 8378563200 8583815059 8792009665 8877689391
##        754        740        754        666        754        753
```

```
sum(is.na(hourly_intensities))
```

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

```
sum(duplicated(hourly_intensities))
```

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

```
summary(hourly_intensities)
```

```
##        Id            ActivityHour       TotalIntensity   AverageIntensity
##  Min.   :1.504e+09   Length:24084       Min.   :  0.00   Min.   :0.00000
##  1st Qu.:2.347e+09   Class :character   1st Qu.:  0.00   1st Qu.:0.00000
##  Median :4.559e+09   Mode  :character   Median :  1.00   Median :0.01667
##  Mean   :4.889e+09                      Mean   : 10.83   Mean   :0.18044
##  3rd Qu.:6.962e+09                      3rd Qu.: 14.00   3rd Qu.:0.23333
##  Max.   :8.878e+09                      Max.   :180.00   Max.   :3.00000
```

```
print("Hourly Steps: ")
```

```
## [1] "Hourly Steps: "
```

```
length(table(hourly_steps$Id))
```

```
## [1] 34
```

```
table(hourly_steps$Id)
```

```
##
## 1503960366 1624580081 1644430081 1844505072 1927972279 2022484408 2026352035
##        744        755        700        751        755        755        754
## 2320127002 2347167796 2873212765 2891001357 3372868164 3977333714 4020332650
```

```
##        755        751        740         12        711        747        749
## 4057192912 4319703577 4445114986 4558609924 4702921684 5553957443 5577150313
##        750        671        754        754        742        752        730
## 6117666160 6290855005 6391747486 6775888955 6962181067 7007744171 7086361926
##        695        521        675        682        754        752        752
## 8053475328 8253242879 8378563200 8583815059 8792009665 8877689391
##        754        740        754        666        754        753
```

```
sum(is.na(hourly_steps))
```

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

```
sum(duplicated(hourly_steps))
```

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

```
summary(hourly_steps)
```

```
##        Id            ActivityHour         StepTotal
##  Min.   :1.504e+09   Length:24084       Min.   :    0.0
##  1st Qu.:2.347e+09   Class :character   1st Qu.:    0.0
##  Median :4.559e+09   Mode  :character   Median :   10.0
##  Mean   :4.889e+09                      Mean   :  286.2
##  3rd Qu.:6.962e+09                      3rd Qu.:  289.0
##  Max.   :8.878e+09                      Max.   :10565.0
```

```
print("Weightlog Info: ")
```

```
## [1] "Weightlog Info: "
```

```
length(table(weightlog_info$Id))
```

```
## [1] 11
```

```
table(weightlog_info$Id)
```

```
##
## 1503960366 1927972279 2347167796 2873212765 2891001357 4445114986 4558609924
##          1          1          1          2          1          1          1
## 4702921684 6962181067 8253242879 8877689391
##          1         14          1          9
```

```
sum(is.na(weightlog_info))
```

```
## [1] 31
```

```
sum(duplicated(weightlog_info))
```

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

```
summary(weightlog_info)
```

```
##        Id                Date              WeightKg       WeightPounds
##  Min.   :1.504e+09   Length:33          Min.   : 53.30   Min.   :117.5
##  1st Qu.:4.703e+09   Class :character   1st Qu.: 61.70   1st Qu.:136.0
##  Median :6.962e+09   Mode  :character   Median : 62.50   Median :137.8
##  Mean   :6.477e+09                      Mean   : 73.44   Mean   :161.9
##  3rd Qu.:8.878e+09                      3rd Qu.: 85.80   3rd Qu.:189.2
##  Max.   :8.878e+09                      Max.   :129.60   Max.   :285.7
##
```

```
##       Fat              BMI        IsManualReport      LogId
##  Min.   :10   Min.   :21.45   Mode :logical   Min.   :1.459e+12
##  1st Qu.:13   1st Qu.:24.10   FALSE:10        1st Qu.:1.460e+12
##  Median :16   Median :24.39   TRUE :23        Median :1.460e+12
##  Mean   :16   Mean   :25.73                   Mean   :1.460e+12
##  3rd Qu.:19   3rd Qu.:25.76                   3rd Qu.:1.460e+12
##  Max.   :22   Max.   :46.17                   Max.   :1.461e+12
##  NA's   :31
```

**Observations** The max total steps is 28497. I'd be interested in seeing the hourly steps for that person because that is quite a few steps - they might live in a walkable city or walk to/from work or something like that. I'd also like to see if it matches up with the max distance of 27.530 and again what the hourly distance looks like for that person. Actually, in seeing that the max very active distance is 21.920, I wonder if this person had their tracker on while running a marathon - which is about 26 miles - so I'm assuming the distances are measured in miles.

Weightlog_info is the only dataset that has any NAs in it, which makes sense given that it has manual entries. They're all in the "Fat" column - which isn't really a big deal because there's no context about the units of the numbers or how they're acquired, so they're less useful anyway.

There are no duplicates in any of the datasets.

How is the max calories in daily_activity 4562? That doesn't seem right. Mean daily calories is 2189, which feels appropriate. Will have to check the number of entries with daily calories above 3000ish. Actually, many of the max values in daily_activity feel a bit high.

933 calories in one hour also seems really high in hourly_calories, especially when the mean is only 94.27.

The max number of steps from hourly_steps is 10565 when the mean is 286.2. 10565/hour = ~176/minute = ~2.9/second. I want to look at the records causing these maxes and determine if they're outliers, which seems likely.

The max weight in kg and lbs line up - 129.60kg = 285.72lbs, which would be 285.7 to keep with significant figures for the weight in lbs (which is interestingly only 1 digit after the decimal place when the kg weights have 2). More than half of the weight reports are noted as manual.

**Checking for Outliers** We're checking for a few things here.

First, how many records in daily_activity have a total calorie amount above 3000 and 3500 and the actual values of the high-value calorie days.

```
daily_activity %>% filter(Calories > 3000)
```

```
## # A tibble: 70 x 15
##            Id ActivityDate TotalSteps TotalDistance TrackerDistance
##         <dbl> <chr>             <dbl>         <dbl>           <dbl>
##  1 1644430081 4/1/2016           4636          3.41            3.41
##  2 1644430081 4/2/2016          20237         14.7            14.7
##  3 1644430081 4/3/2016          12912          9.41            9.41
##  4 1644430081 4/5/2016           9921          7.21            7.21
##  5 1644430081 4/7/2016          11166          8.12            8.12
##  6 1644430081 4/9/2016          13840         10.1            10.1
##  7 2022484408 4/7/2016          18247         13.8            13.8
##  8 2891001357 4/1/2016              0          0               0
##  9 4020332650 3/15/2016          5906          4.23            4.23
## 10 4020332650 3/16/2016         12483          8.99            8.99
## # i 60 more rows
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
```

```
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

```
daily_activity %>% filter(Calories > 3500)
```

```
## # A tibble: 34 x 15
##            Id ActivityDate TotalSteps TotalDistance TrackerDistance
##         <dbl> <chr>              <dbl>         <dbl>           <dbl>
##  1 1644430081 4/2/2016           20237         14.7            14.7
##  2 2891001357 4/1/2016               0          0               0
##  3 4020332650 3/16/2016          12483          8.99            8.99
##  4 4020332650 3/17/2016           8940          6.41            6.41
##  5 4020332650 3/20/2016          10330          7.41            7.41
##  6 4020332650 3/25/2016           5563          3.99            3.99
##  7 4020332650 3/27/2016           7144          5.12            5.12
##  8 4020332650 3/28/2016           2106          1.51            1.51
##  9 4020332650 4/8/2016           10480          7.51            7.51
## 10 4702921684 4/9/2016           14002         11.4            11.4
## # i 24 more rows
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

```
high_daily_calories <- daily_activity %>% filter(Calories > 3500)
summary(high_daily_calories)
```

```
##        Id            ActivityDate         TotalSteps    TotalDistance
##  Min.   :1.644e+09   Length:34          Min.   :    0   Min.   : 0.000
##  1st Qu.:4.191e+09   Class :character   1st Qu.: 9180   1st Qu.: 7.067
##  Median :6.776e+09   Mode  :character   Median :12122   Median : 8.245
##  Mean   :6.568e+09                      Mean   :13511   Mean   :10.526
##  3rd Qu.:8.379e+09                      3rd Qu.:16168   3rd Qu.:11.330
##  Max.   :8.878e+09                      Max.   :28497   Max.   :27.530
##  TrackerDistance  LoggedActivitiesDistance VeryActiveDistance
##  Min.   : 0.000   Min.   :0.0000           Min.   : 0.000
##  1st Qu.: 7.067   1st Qu.:0.0000           1st Qu.: 0.460
##  Median : 8.245   Median :0.0000           Median : 2.490
##  Mean   :10.526   Mean   :0.5207           Mean   : 4.562
##  3rd Qu.:11.330   3rd Qu.:0.0000           3rd Qu.: 5.923
##  Max.   :27.530   Max.   :4.8280           Max.   :21.920
##  ModeratelyActiveDistance LightActiveDistance SedentaryActiveDistance
##  Min.   :0.000            Min.   :0.000       Min.   :0.000000
##  1st Qu.:0.205            1st Qu.:2.265       1st Qu.:0.000000
##  Median :0.700            Median :4.525       Median :0.000000
##  Mean   :1.008            Mean   :4.219       Mean   :0.002941
##  3rd Qu.:1.188            3rd Qu.:6.128       3rd Qu.:0.000000
##  Max.   :6.400            Max.   :8.620       Max.   :0.040000
##  VeryActiveMinutes FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes
##  Min.   :  0.00    Min.   :  0.00      Min.   :  0.0        Min.   : 407.0
##  1st Qu.: 26.50    1st Qu.: 11.75      1st Qu.:142.8        1st Qu.: 701.0
##  Median : 72.50    Median : 19.50      Median :201.5        Median : 893.0
```

```
##  Mean   : 68.41    Mean   : 52.65      Mean   :194.1        Mean   : 909.1
##  3rd Qu.:103.50    3rd Qu.: 45.75      3rd Qu.:259.8        3rd Qu.:1057.2
##  Max.   :202.00    Max.   :660.00      Max.   :381.0        Max.   :1440.0
##     Calories
##  Min.   :3510
##  1st Qu.:3699
##  Median :3834
##  Mean   :3903
##  3rd Qu.:4038
##  Max.   :4562
```

```
print(high_daily_calories[order(high_daily_calories$Calories, decreasing = TRUE), ])
```

```
## # A tibble: 34 x 15
##            Id ActivityDate TotalSteps TotalDistance TrackerDistance
##         <dbl> <chr>              <dbl>         <dbl>           <dbl>
##  1 2891001357 4/1/2016               0          0               0
##  2 8877689391 4/10/2016          28497         27.5            27.5
##  3 5577150313 4/2/2016           14873         11.1            11.1
##  4 6775888955 4/5/2016            9348          6.70            6.70
##  5 8378563200 4/4/2016           13935         11.1            11.1
##  6 8877689391 4/2/2016           27572         23.4            23.4
##  7 8877689391 4/8/2016           23014         20.4            20.4
##  8 8378563200 4/5/2016           12846         10.2            10.2
##  9 8877689391 4/6/2016           24136         20.9            20.9
## 10 4020332650 4/8/2016           10480          7.51            7.51
## # i 24 more rows
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

There are 70 records with a Calories value above 3000, 34 above 3500, and 12 above 4000, so 4562 is not as much of an outlier as I suspected. They also have pretty high tracked/total distances and steps, added more confidence to the validity of the values.

Then, how many hourly records fall into specified calorie ranges.

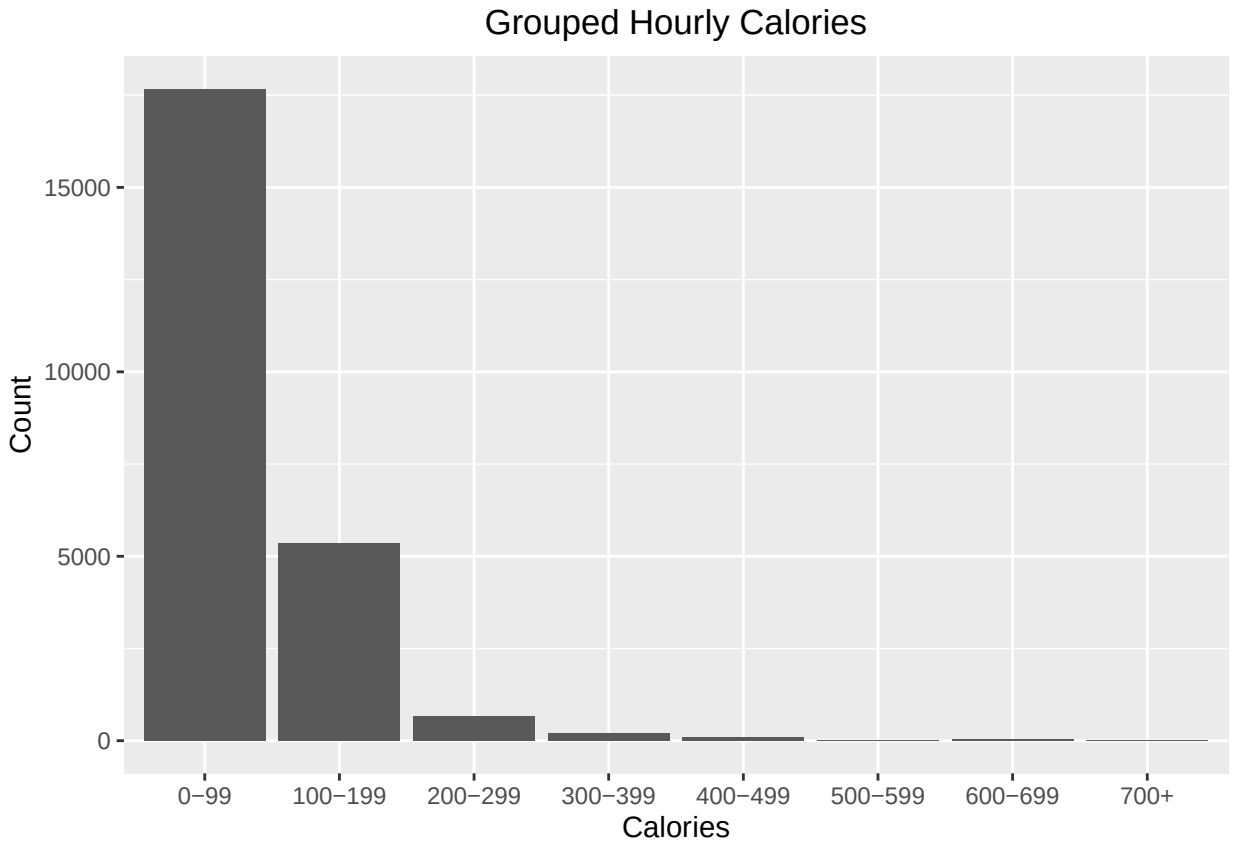
```
print(hourly_calories[order(hourly_calories$Calories, decreasing = TRUE), ])
```

```
## # A tibble: 24,084 x 3
##            Id ActivityHour          Calories
##         <dbl> <chr>                    <dbl>
##  1 8877689391 4/10/2016 11:00:00 AM      933
##  2 8877689391 3/27/2016 3:00:00 PM       916
##  3 8877689391 3/27/2016 4:00:00 PM       906
##  4 8877689391 3/12/2016 2:00:00 PM       856
##  5 8877689391 3/23/2016 6:00:00 PM       848
##  6 8877689391 4/2/2016 4:00:00 PM        826
##  7 8877689391 4/4/2016 7:00:00 PM        800
##  8 8877689391 3/17/2016 6:00:00 PM       760
##  9 8877689391 3/16/2016 6:00:00 PM       734
## 10 8877689391 3/22/2016 6:00:00 PM       716
## # i 24,074 more rows
```

```
hourly_cal_buckets <- cut(hourly_calories$Calories, breaks = c(0, 100, 200, 300, 400, 500, 600, 700, Inf
cal_bucket_df <- as.data.frame(table(hourly_cal_buckets))
colnames(cal_bucket_df) <- c("Calories", "Count")
print(cal_bucket_df)
```

```
##   Calories Count
## 1     0-99 17674
## 2  100-199  5364
## 3  200-299   667
## 4  300-399   210
## 5  400-499   103
## 6  500-599    26
## 7  600-699    28
## 8     700+    12
```

```
ggplot(cal_bucket_df, aes(x = Calories, y = Count)) +
  geom_col() +
  ggtitle("Grouped Hourly Calories") + xlab("Calories") + ylab("Count") +
  theme(plot.title = element_text(hjust = 0.5))
```




There are 66 records with a value over 500 calories in hourly_calories and 12 with over 700 - including 3 over 900 - so 993 also isn't as much of an outlier as I expected.

Finally, the daily activity records for the users with the maximum value for hourly calories and hourly steps, starting with getting their Ids.

```
max_hourly_calories <- hourly_calories[hourly_calories$Calories == 933,]
print(max_hourly_calories)
```

```
## # A tibble: 1 x 3
##           Id ActivityHour          Calories
##        <dbl> <chr>                    <dbl>
## 1 8877689391 4/10/2016 11:00:00 AM      933
```

```
max_steps <- hourly_steps[hourly_steps$StepTotal == 10565,]
print(max_steps)
```

```
## # A tibble: 1 x 3
##           Id ActivityHour          StepTotal
##        <dbl> <chr>                     <dbl>
## 1 8877689391 4/10/2016 11:00:00 AM     10565
```

Both the max hourly calories and max hourly steps belong to the same user on the same hour/day, so now let's look at that record in daily activity.

```
print(daily_activity[daily_activity$Id == 8877689391 & daily_activity$ActivityDate == "4/10/2016",])
```

```
## # A tibble: 1 x 15
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
##        <dbl> <chr>             <dbl>         <dbl>           <dbl>
## 1 8877689391 4/10/2016         28497          27.5            27.5
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

The record with a calorie value of 4562 has a total distance value of 27.53 and a VeryActiveMinutes value of 128, which would be just over 2 hours of being "very active". It also has 46 fairly active minutes and 211 lightly active minutes, which add up to 385 total minutes with the very active minutes included - almost 6.5 hours. The average marathon time is around 4 hours, so I believe this further supports the hypothesis that this user ran a marathon with their device on.

And perhaps unsurprisingly, the record with an hourly calories of 933 is the same record as the one with a step total of 10565 - the user that probably ran a marathon.

```
print(weightlog_info)
```

```
## # A tibble: 33 x 8
##            Id Date      WeightKg WeightPounds   Fat   BMI IsManualReport   LogId
##         <dbl> <chr>        <dbl>        <dbl> <dbl> <dbl> <lgl>            <dbl>
##  1 1503960366 4/5/2016~     53.3         118.    22  23.0 TRUE           1.46e12
##  2 1927972279 4/10/201~    130.          286.    NA  46.2 FALSE          1.46e12
##  3 2347167796 4/3/2016~     63.4         140.    10  24.8 TRUE           1.46e12
##  4 2873212765 4/6/2016~     56.7         125.    NA  21.5 TRUE           1.46e12
##  5 2873212765 4/7/2016~     57.2         126.    NA  21.6 TRUE           1.46e12
##  6 2891001357 4/5/2016~     88.4         195.    NA  25.0 TRUE           1.46e12
##  7 4445114986 3/30/201~     92.4         204.    NA  35.0 TRUE           1.46e12
##  8 4558609924 4/8/2016~     69.4         153.    NA  27.1 TRUE           1.46e12
##  9 4702921684 4/4/2016~     99.7         220.    NA  26.1 TRUE           1.46e12
## 10 6962181067 3/30/201~     61.5         136.    NA  24.0 TRUE           1.46e12
## # i 23 more rows
```

## PROCESS

**Revisiting the first main question:**

Question 1: What are some trends in smart device usage?

We have the following data: Datetime Distances traveled/moved Steps taken Time spent at various activity levels Calories Activity intensity Weight (probably unreliable)

So for each of the thoughts I posed, I'll look at the data as such: 1. When during the day are people using smart devices most active? Check highest calories, intensities, steps - Datetime, steps taken, activity intensities, calories

2. How much are people walking, when, and at what intensities? Compare highest steps with intensity and time

- Datetime, steps taken, activity intensities (can maybe be bundled with 1.)

3a. How many weightlog reports are manual vs automatic? - This was answered in the summary for the weightLoginfo dataset - 10 automatic and 23 manual

3b. Do the reports for the same people show any interesting trends? - There are only multiple reports for 3 of the 11 people represented in weightLoginfo and one of those only has 2 reports. I'll check the other two with multiple reports to be thorough but don't expect to see too much.

3c. Does any weight/fat loss or gain coincide with higher or lower levels of activity/intensity? Compare changes in weightlog with corresponding intensities/steps/calories - weight, steps, calories, averageintensity

4. Does it seem like people are wearing their smart devices all day or only when doing activities? Check steps/calories for an entire day as well as activity/sedentary duration in daily_activity

- sedentary minutes, active minutes

5. Do people tend to sleep better after days with higher amounts/intensities of activity?

- Ended up not going with this due to unclear sleep data

6. What are the maxes (and mins?) for each individual and do they line up/make sense?

- ID, calories, steps, distance (pretty much everything)

### Data Manipulation

**Date/Time Split and Alignment** I want to split the various date columns so that there's a column for date and a column for time so we can match up dates in the analysis

Split the datetime columns into date and time and confirm that it was done correctly

```
hourly_calories <- separate(hourly_calories, ActivityHour, into = c('Date', 'Time', 'AMPM'), sep=' ', FA
hourly_calories <- unite(hourly_calories, 'Time', Time, AMPM, sep=' ', remove = TRUE)

hourly_intensities <- separate(hourly_intensities, ActivityHour, into = c('Date', 'Time', 'AMPM'), sep=
hourly_intensities <- unite(hourly_intensities, 'Time', Time, AMPM, sep=' ', remove = TRUE)

hourly_steps <- separate(hourly_steps, ActivityHour, into = c('Date', 'Time', 'AMPM'), sep=' ', FALSE)
hourly_steps <- unite(hourly_steps, 'Time', Time, AMPM, sep=' ', remove = TRUE)

weightlog_info <- separate(weightlog_info, Date, into = c('Date', 'Time', 'AMPM'), sep=' ', FALSE)
weightlog_info <- unite(weightlog_info, 'Time', Time, AMPM, sep=' ', remove = TRUE)

head(hourly_calories)
```

```
## # A tibble: 6 x 5
##           Id ActivityHour          Date      Time        Calories
##        <dbl> <chr>                 <chr>     <chr>          <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM 3/12/2016 12:00:00 AM       48
## 2 1503960366 3/12/2016 1:00:00 AM  3/12/2016 1:00:00 AM        48
## 3 1503960366 3/12/2016 2:00:00 AM  3/12/2016 2:00:00 AM        48
## 4 1503960366 3/12/2016 3:00:00 AM  3/12/2016 3:00:00 AM        48
## 5 1503960366 3/12/2016 4:00:00 AM  3/12/2016 4:00:00 AM        48
## 6 1503960366 3/12/2016 5:00:00 AM  3/12/2016 5:00:00 AM        48
```

```
head(hourly_intensities)
```

```
## # A tibble: 6 x 6
##           Id ActivityHour          Date    Time  TotalIntensity AverageIntensity
##        <dbl> <chr>                 <chr>   <chr>          <dbl>             <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM 3/12/2~ 12:0~             0                 0
## 2 1503960366 3/12/2016 1:00:00 AM  3/12/2~ 1:00~             0                 0
## 3 1503960366 3/12/2016 2:00:00 AM  3/12/2~ 2:00~             0                 0
## 4 1503960366 3/12/2016 3:00:00 AM  3/12/2~ 3:00~             0                 0
## 5 1503960366 3/12/2016 4:00:00 AM  3/12/2~ 4:00~             0                 0
## 6 1503960366 3/12/2016 5:00:00 AM  3/12/2~ 5:00~             0                 0
```

```
head(hourly_steps)
```

```
## # A tibble: 6 x 5
##           Id ActivityHour          Date      Time        StepTotal
##        <dbl> <chr>                 <chr>     <chr>           <dbl>
## 1 1503960366 3/12/2016 12:00:00 AM 3/12/2016 12:00:00 AM         0
## 2 1503960366 3/12/2016 1:00:00 AM  3/12/2016 1:00:00 AM          0
## 3 1503960366 3/12/2016 2:00:00 AM  3/12/2016 2:00:00 AM          0
## 4 1503960366 3/12/2016 3:00:00 AM  3/12/2016 3:00:00 AM          0
## 5 1503960366 3/12/2016 4:00:00 AM  3/12/2016 4:00:00 AM          0
## 6 1503960366 3/12/2016 5:00:00 AM  3/12/2016 5:00:00 AM          0
```

```
head(weightlog_info)
```

```
## # A tibble: 6 x 9
##          Id Date  Time  WeightKg WeightPounds   Fat   BMI IsManualReport   LogId
##       <dbl> <chr> <chr>    <dbl>        <dbl> <dbl> <dbl> <lgl>            <dbl>
## 1    1.50e9 4/5/~ 11:5~     53.3         118.    22  23.0 TRUE           1.46e12
## 2    1.93e9 4/10~ 6:33~    130.          286.    NA  46.2 FALSE          1.46e12
## 3    2.35e9 4/3/~ 11:5~     63.4         140.    10  24.8 TRUE           1.46e12
## 4    2.87e9 4/6/~ 11:5~     56.7         125.    NA  21.5 TRUE           1.46e12
## 5    2.87e9 4/7/~ 11:5~     57.2         126.    NA  21.6 TRUE           1.46e12
## 6    2.89e9 4/5/~ 11:5~     88.4         195.    NA  25.0 TRUE           1.46e12
```

**Table Manipulation** Now that we've separated the date and time for later manipulation, let's combine the hourly datasets and check it.

```
hourly_cal_step <- inner_join(hourly_calories, hourly_steps, by = c("Id" = "Id", "ActivityHour" = "Acti
combined_hourly <- inner_join(hourly_cal_step, hourly_intensities, by = c("Id" = "Id", "ActivityHour" =
print(combined_hourly)
```

```
## # A tibble: 24,084 x 8
##            Id ActivityHour         Date  Time  Calories StepTotal TotalIntensity
##         <dbl> <chr>                <chr> <chr>    <dbl>     <dbl>          <dbl>
##  1 1503960366 3/12/2016 12:00:00 ~ 3/12~ 12:0~       48         0              0
```

```
##  2 1503960366 3/12/2016 1:00:00 AM 3/12~ 1:00~       48         0              0
##  3 1503960366 3/12/2016 2:00:00 AM 3/12~ 2:00~       48         0              0
##  4 1503960366 3/12/2016 3:00:00 AM 3/12~ 3:00~       48         0              0
##  5 1503960366 3/12/2016 4:00:00 AM 3/12~ 4:00~       48         0              0
##  6 1503960366 3/12/2016 5:00:00 AM 3/12~ 5:00~       48         0              0
##  7 1503960366 3/12/2016 6:00:00 AM 3/12~ 6:00~       48         0              0
##  8 1503960366 3/12/2016 7:00:00 AM 3/12~ 7:00~       48         0              0
##  9 1503960366 3/12/2016 8:00:00 AM 3/12~ 8:00~       48         0              0
## 10 1503960366 3/12/2016 9:00:00 AM 3/12~ 9:00~       49         8              1
## # i 24,074 more rows
## # i 1 more variable: AverageIntensity <dbl>
```

```
summary(combined_hourly)
```

```
##        Id            ActivityHour           Date               Time
##  Min.   :1.504e+09   Length:24084       Length:24084       Length:24084
##  1st Qu.:2.347e+09   Class :character   Class :character   Class :character
##  Median :4.559e+09   Mode  :character   Mode  :character   Mode  :character
##  Mean   :4.889e+09
##  3rd Qu.:6.962e+09
##  Max.   :8.878e+09
##     Calories        StepTotal       TotalIntensity   AverageIntensity
##  Min.   : 42.00   Min.   :    0.0   Min.   :  0.00   Min.   :0.00000
##  1st Qu.: 61.00   1st Qu.:    0.0   1st Qu.:  0.00   1st Qu.:0.00000
##  Median : 77.00   Median :   10.0   Median :  1.00   Median :0.01667
##  Mean   : 94.27   Mean   :  286.2   Mean   : 10.83   Mean   :0.18044
##  3rd Qu.:104.00   3rd Qu.:  289.0   3rd Qu.: 14.00   3rd Qu.:0.23333
##  Max.   :933.00   Max.   :10565.0   Max.   :180.00   Max.   :3.00000
```

```
table(combined_hourly$Id)
```

```
##
## 1503960366 1624580081 1644430081 1844505072 1927972279 2022484408 2026352035
##        744        755        700        751        755        755        754
## 2320127002 2347167796 2873212765 2891001357 3372868164 3977333714 4020332650
##        755        751        740         12        711        747        749
## 4057192912 4319703577 4445114986 4558609924 4702921684 5553957443 5577150313
##        750        671        754        754        742        752        730
## 6117666160 6290855005 6391747486 6775888955 6962181067 7007744171 7086361926
##        695        521        675        682        754        752        752
## 8053475328 8253242879 8378563200 8583815059 8792009665 8877689391
##        754        740        754        666        754        753
```

```
length(table(combined_hourly$Id))
```

```
## [1] 34
```

Now let's group this combined table by day so we can compare it to the daily_activity measures.

```
grouped_combined_hourly <- combined_hourly %>%
  group_by(Id, Date) %>%
  summarize(total_calories = sum(Calories), total_steps = sum(StepTotal), avg_total_intensity = mean(To
```

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

```
print(grouped_combined_hourly)
```

```
## # A tibble: 1,021 x 6
## # Groups:   Id [34]
##        Id Date  total_calories total_steps avg_total_intensity avg_avg_intensity
##     <dbl> <chr>          <dbl>       <dbl>               <dbl>             <dbl>
##  1 1.50e9 3/12~           2228       19675                24.8             0.414
##  2 1.50e9 3/13~           2100       17106                23.2             0.387
##  3 1.50e9 3/14~           1830       10023                15.4             0.256
##  4 1.50e9 3/15~           2111       15384                20.6             0.344
##  5 1.50e9 3/16~           1967       13498                18.1             0.301
##  6 1.50e9 3/17~           2039       14027                18.1             0.302
##  7 1.50e9 3/18~           2002       14544                17.2             0.287
##  8 1.50e9 3/19~           2057       15424                19.7             0.328
##  9 1.50e9 3/20~           2096       15128                19.2             0.320
## 10 1.50e9 3/21~           1846       10020                14.3             0.239
## # i 1,011 more rows
```

```
print(daily_activity)
```

```
## # A tibble: 457 x 15
##            Id ActivityDate TotalSteps TotalDistance TrackerDistance
##         <dbl> <chr>             <dbl>         <dbl>           <dbl>
##  1 1503960366 3/25/2016         11004          7.11            7.11
##  2 1503960366 3/26/2016         17609         11.6            11.6
##  3 1503960366 3/27/2016         12736          8.53            8.53
##  4 1503960366 3/28/2016         13231          8.93            8.93
##  5 1503960366 3/29/2016         12041          7.85            7.85
##  6 1503960366 3/30/2016         10970          7.16            7.16
##  7 1503960366 3/31/2016         12256          7.86            7.86
##  8 1503960366 4/1/2016          12262          7.87            7.87
##  9 1503960366 4/2/2016          11248          7.25            7.25
## 10 1503960366 4/3/2016          10016          6.37            6.37
## # i 447 more rows
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

**Table Verification and Observations** Based on the top rows, it does appear that the sum of the hourly calories and steps matches the daily calories and steps in the daily activity dataset. There are some instances where the total is 1 or 2 calories off (1943 vs 1944 on 3/27/2016 for 1503960366, for example), but I think we can assume this is from rounding.

Now we'll check the combined table to make sure there are no missing or duplicate values.

```
print("Number of missing values")
```

```
## [1] "Number of missing values"
```

```
sum(is.na(grouped_combined_hourly))
```

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

```
print("Number of duplicated values")
```

```
## [1] "Number of duplicated values"
```

```r
sum(duplicated(grouped_combined_hourly))
```

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

```r
print("Summary of grouped data")
```

```
## [1] "Summary of grouped data"
```

```r
summary(grouped_combined_hourly)
```

```
##        Id                Date           total_calories  total_steps
##  Min.   :1.504e+09   Length:1021        Min.   : 164   Min.   :    0
##  1st Qu.:2.347e+09   Class :character   1st Qu.:1776   1st Qu.: 2030
##  Median :4.559e+09   Mode  :character   Median :2100   Median : 6248
##  Mean   :4.888e+09                      Mean   :2224   Mean   : 6752
##  3rd Qu.:6.962e+09                      3rd Qu.:2666   3rd Qu.:10461
##  Max.   :8.878e+09                      Max.   :4776   Max.   :37322
##  avg_total_intensity avg_avg_intensity
##  Min.   : 0.00       Min.   :0.0000
##  1st Qu.: 4.50       1st Qu.:0.0750
##  Median :11.17       Median :0.1861
##  Mean   :10.76       Mean   :0.1793
##  3rd Qu.:16.08       3rd Qu.:0.2681
##  Max.   :41.29       Max.   :0.6882
```

```r
print("Summary of base daily data")
```

```
## [1] "Summary of base daily data"
```

```r
summary(daily_activity)
```

```
##        Id            ActivityDate         TotalSteps    TotalDistance
##  Min.   :1.504e+09   Length:457         Min.   :    0   Min.   : 0.000
##  1st Qu.:2.347e+09   Class :character   1st Qu.: 1988   1st Qu.: 1.410
##  Median :4.057e+09   Mode  :character   Median : 5986   Median : 4.090
##  Mean   :4.629e+09                      Mean   : 6547   Mean   : 4.664
##  3rd Qu.:6.392e+09                      3rd Qu.:10198   3rd Qu.: 7.160
##  Max.   :8.878e+09                      Max.   :28497   Max.   :27.530
##  TrackerDistance LoggedActivitiesDistance VeryActiveDistance
##  Min.   : 0.00   Min.   :0.0000           Min.   : 0.000
##  1st Qu.: 1.28   1st Qu.:0.0000           1st Qu.: 0.000
##  Median : 4.09   Median :0.0000           Median : 0.000
##  Mean   : 4.61   Mean   :0.1794           Mean   : 1.181
##  3rd Qu.: 7.11   3rd Qu.:0.0000           3rd Qu.: 1.310
##  Max.   :27.53   Max.   :6.7271           Max.   :21.920
##  ModeratelyActiveDistance LightActiveDistance SedentaryActiveDistance
##  Min.   :0.0000           Min.   : 0.00       Min.   :0.000000
##  1st Qu.:0.0000           1st Qu.: 0.87       1st Qu.:0.000000
##  Median :0.0200           Median : 2.93       Median :0.000000
##  Mean   :0.4786           Mean   : 2.89       Mean   :0.001904
##  3rd Qu.:0.6700           3rd Qu.: 4.46       3rd Qu.:0.000000
##  Max.   :6.4000           Max.   :12.51       Max.   :0.100000
##  VeryActiveMinutes FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes
##  Min.   :  0.00    Min.   :  0.00      Min.   :  0.0        Min.   :  32.0
##  1st Qu.:  0.00    1st Qu.:  0.00      1st Qu.: 64.0        1st Qu.: 728.0
##  Median :  0.00    Median :  1.00      Median :181.0        Median :1057.0
##  Mean   : 16.62    Mean   : 13.07      Mean   :170.1        Mean   : 995.3
```

```
##  3rd Qu.: 25.00    3rd Qu.: 16.00      3rd Qu.:257.0        3rd Qu.:1285.0
##  Max.   :202.00    Max.   :660.00      Max.   :720.0        Max.   :1440.0
##     Calories
##  Min.   :   0
##  1st Qu.:1776
##  Median :2062
##  Mean   :2189
##  3rd Qu.:2667
##  Max.   :4562
```

But looking at the summaries, the maxes don't line up. The max total calories is 4776 in the combined dataset vs 4562 in daily activity and the max steps is 37322 vs 28497. Let's take a look at those records.

```
max_steps <- grouped_combined_hourly[grouped_combined_hourly$total_steps==37322,]
print(max_steps)
```

```
## # A tibble: 1 x 6
## # Groups:   Id [1]
##        Id Date  total_calories total_steps avg_total_intensity avg_avg_intensity
##     <dbl> <chr>          <dbl>       <dbl>               <dbl>             <dbl>
## 1  8.05e9 3/13~           4528       37322                40.6             0.677
```

```
max_calories <- grouped_combined_hourly[grouped_combined_hourly$total_calories==4776,]
print(max_calories)
```

```
## # A tibble: 1 x 6
## # Groups:   Id [1]
##        Id Date  total_calories total_steps avg_total_intensity avg_avg_intensity
##     <dbl> <chr>          <dbl>       <dbl>               <dbl>             <dbl>
## 1  5.58e9 3/12~           4776       19450                41.3             0.688
```

So the Id and date of the person with 37322 steps is 8053475328 on 3/13/2016 and the person with 4776 calories is 5577150313 on 3/12/2016. Looking at their entries in the corresponding hourly dataset and daily activity for that day:

```
step_check_hourly <- hourly_steps[hourly_steps$Id==8053475328 & hourly_steps$Date=="3/13/2016",]
print(step_check_hourly)
```

```
## # A tibble: 24 x 5
##            Id ActivityHour          Date      Time        StepTotal
##         <dbl> <chr>                 <chr>     <chr>           <dbl>
##  1 8053475328 3/13/2016 12:00:00 AM 3/13/2016 12:00:00 AM       330
##  2 8053475328 3/13/2016 1:00:00 AM  3/13/2016 1:00:00 AM        268
##  3 8053475328 3/13/2016 2:00:00 AM  3/13/2016 2:00:00 AM         19
##  4 8053475328 3/13/2016 3:00:00 AM  3/13/2016 3:00:00 AM          0
##  5 8053475328 3/13/2016 4:00:00 AM  3/13/2016 4:00:00 AM          0
##  6 8053475328 3/13/2016 5:00:00 AM  3/13/2016 5:00:00 AM          0
##  7 8053475328 3/13/2016 6:00:00 AM  3/13/2016 6:00:00 AM          0
##  8 8053475328 3/13/2016 7:00:00 AM  3/13/2016 7:00:00 AM          0
##  9 8053475328 3/13/2016 8:00:00 AM  3/13/2016 8:00:00 AM          0
## 10 8053475328 3/13/2016 9:00:00 AM  3/13/2016 9:00:00 AM          0
## # i 14 more rows
```

```
print(sum(step_check_hourly$StepTotal))
```

```
## [1] 37322
```

```r
daily_check_1 <- daily_activity[daily_activity$Id==8053475328,]
print(daily_check_1)
```

```
## # A tibble: 11 x 15
##            Id ActivityDate TotalSteps TotalDistance TrackerDistance
##         <dbl> <chr>              <dbl>         <dbl>           <dbl>
##  1 8053475328 4/2/2016           20188        15.6            15.6
##  2 8053475328 4/3/2016           25701        20.1            20.1
##  3 8053475328 4/4/2016           17395        13.2            13.2
##  4 8053475328 4/5/2016           17167        13.4            13.4
##  5 8053475328 4/6/2016           16435        12.4            12.4
##  6 8053475328 4/7/2016           17078        13.2            13.2
##  7 8053475328 4/8/2016           11693         9.54            9.54
##  8 8053475328 4/9/2016           11159         9.14            9.14
##  9 8053475328 4/10/2016          10118         7.73            7.73
## 10 8053475328 4/11/2016          16064        12.7            12.7
## 11 8053475328 4/12/2016            290         0.210           0.210
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

```r
calorie_check_hourly <- hourly_calories[hourly_calories$Id==5577150313 & hourly_calories$Date=="3/12/20
print(calorie_check_hourly)
```

```
## # A tibble: 24 x 5
##            Id ActivityHour          Date      Time        Calories
##         <dbl> <chr>                 <chr>     <chr>          <dbl>
##  1 5577150313 3/12/2016 12:00:00 AM 3/12/2016 12:00:00 AM       78
##  2 5577150313 3/12/2016 1:00:00 AM  3/12/2016 1:00:00 AM        86
##  3 5577150313 3/12/2016 2:00:00 AM  3/12/2016 2:00:00 AM        80
##  4 5577150313 3/12/2016 3:00:00 AM  3/12/2016 3:00:00 AM        78
##  5 5577150313 3/12/2016 4:00:00 AM  3/12/2016 4:00:00 AM        77
##  6 5577150313 3/12/2016 5:00:00 AM  3/12/2016 5:00:00 AM        80
##  7 5577150313 3/12/2016 6:00:00 AM  3/12/2016 6:00:00 AM       418
##  8 5577150313 3/12/2016 7:00:00 AM  3/12/2016 7:00:00 AM        79
##  9 5577150313 3/12/2016 8:00:00 AM  3/12/2016 8:00:00 AM       309
## 10 5577150313 3/12/2016 9:00:00 AM  3/12/2016 9:00:00 AM       394
## # i 14 more rows
```

```r
print(sum(calorie_check_hourly$Calories))
```

```
## [1] 4776
```

```r
daily_check_2 <- daily_activity[daily_activity$Id==5577150313,]
print(daily_check_2)
```

```
## # A tibble: 11 x 15
##            Id ActivityDate TotalSteps TotalDistance TrackerDistance
##         <dbl> <chr>              <dbl>         <dbl>           <dbl>
##  1 5577150313 4/1/2016           10461          7.87            7.87
##  2 5577150313 4/2/2016           14873         11.1            11.1
##  3 5577150313 4/3/2016            9917          7.41            7.41
##  4 5577150313 4/4/2016            7401          5.56            5.56
##  5 5577150313 4/5/2016            8964          6.70            6.70
```

```
##  6 5577150313 4/6/2016          11080          8.30           8.30
##  7 5577150313 4/7/2016           4499          3.36           3.36
##  8 5577150313 4/8/2016           4363          3.26           3.26
##  9 5577150313 4/9/2016          10494          7.84           7.84
## 10 5577150313 4/10/2016          9776          7.38           7.38
## 11 5577150313 4/11/2016          2862          2.14           2.14
## # i 10 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

Both of these values seem to be valid, and doing this check made me realize that the dailyActivity_merged file and the hourlyCalories/Steps files don't cover the exact same date range. While some users have March data in dailyActivity_merged, the two users with the actual max steps and calories for a single day do not - though data for these days does exist in the hourly datasets.

## ANALYZE

Now we can work towards answering the remaining questions. These are the questions for reference: 1. When during the day are people using smart devices most active?

2. How much are people walking, when, and at what intensities?

3b. Do the reports for the same people show any interesting trends?

3c. Does any weight/fat loss or gain coincide with higher or lower levels of activity/intensity?

4. Does it seem like people are wearing their smart devices all day or only when doing activities?

5. Do people tend to sleep better after days with higher amounts/intensities of activity?

6. What are the maxes (and mins?) for each individual and do they line up/make sense?

### 1a. When during the day are people using smart devices most active? Check highest calories, intensities, steps

Let's make another grouped dataset but grouped on the hour of the day this time. And we're looking for the average activity for all users so we only need to group by the hour, using an added column - index - to sort the grouped data appropriately.

```
hour_based_grouping <- combined_hourly %>%
  group_by(Time) %>%
  summarize(avg_calories = mean(Calories), max_calories = max(Calories), avg_steps = mean(StepTotal), ma
hour_based_grouping$index <- c(10, 22, 11, 23, 1, 12, 2, 13, 3, 14, 4, 15, 5, 16, 6, 17, 7, 18, 8, 19, 9
print(hour_based_grouping)
```

```
## # A tibble: 24 x 9
##    Time        avg_calories max_calories avg_steps max_steps avg_total_intensity
##    <chr>              <dbl>        <dbl>     <dbl>     <dbl>               <dbl>
##  1 10:00:00 AM        106.           538     437.       6163              15.9
##  2 10:00:00 PM         84.8          612     173.       7613               7.41
##  3 11:00:00 AM        108.           933     453.      10565              16.6
##  4 11:00:00 PM         77.0          424     103.       4067               4.57
##  5 12:00:00 AM         72.4          324      44.4      2854               2.41
##  6 12:00:00 PM        114.           708     521.       7163              18.5
##  7 1:00:00 AM          69.6          231      20.5       809               1.32
##  8 1:00:00 PM         109.           534     458.       5840              16.3
##  9 2:00:00 AM          68.1          149      10.6       586               0.760
```

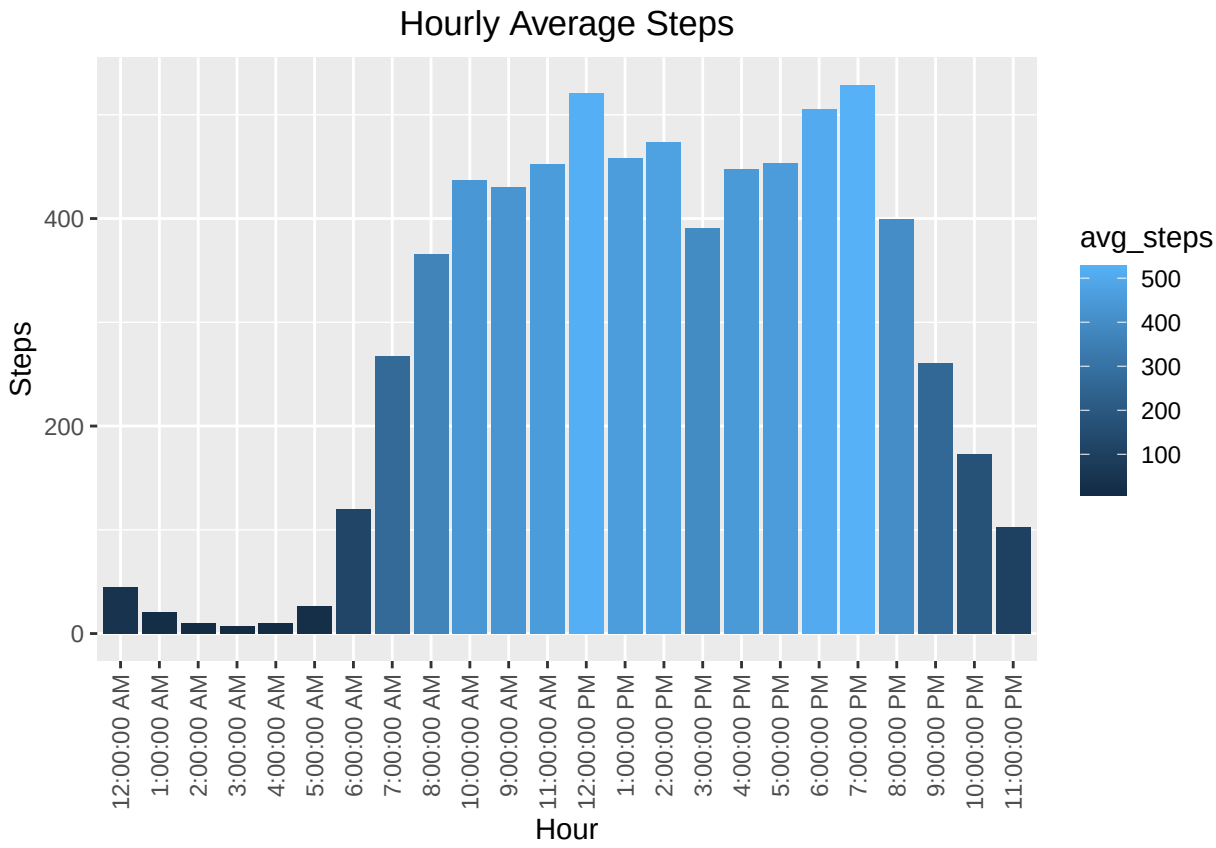
```
## 10 2:00:00 PM         111.           856     474.       9149              17.0
## # i 14 more rows
## # i 3 more variables: max_total_intensity <dbl>, avg_avg_intensity <dbl>,
## #   index <dbl>
```

```
sorted_hour_based_grouping <- arrange(hour_based_grouping, index)
print(sorted_hour_based_grouping)
```

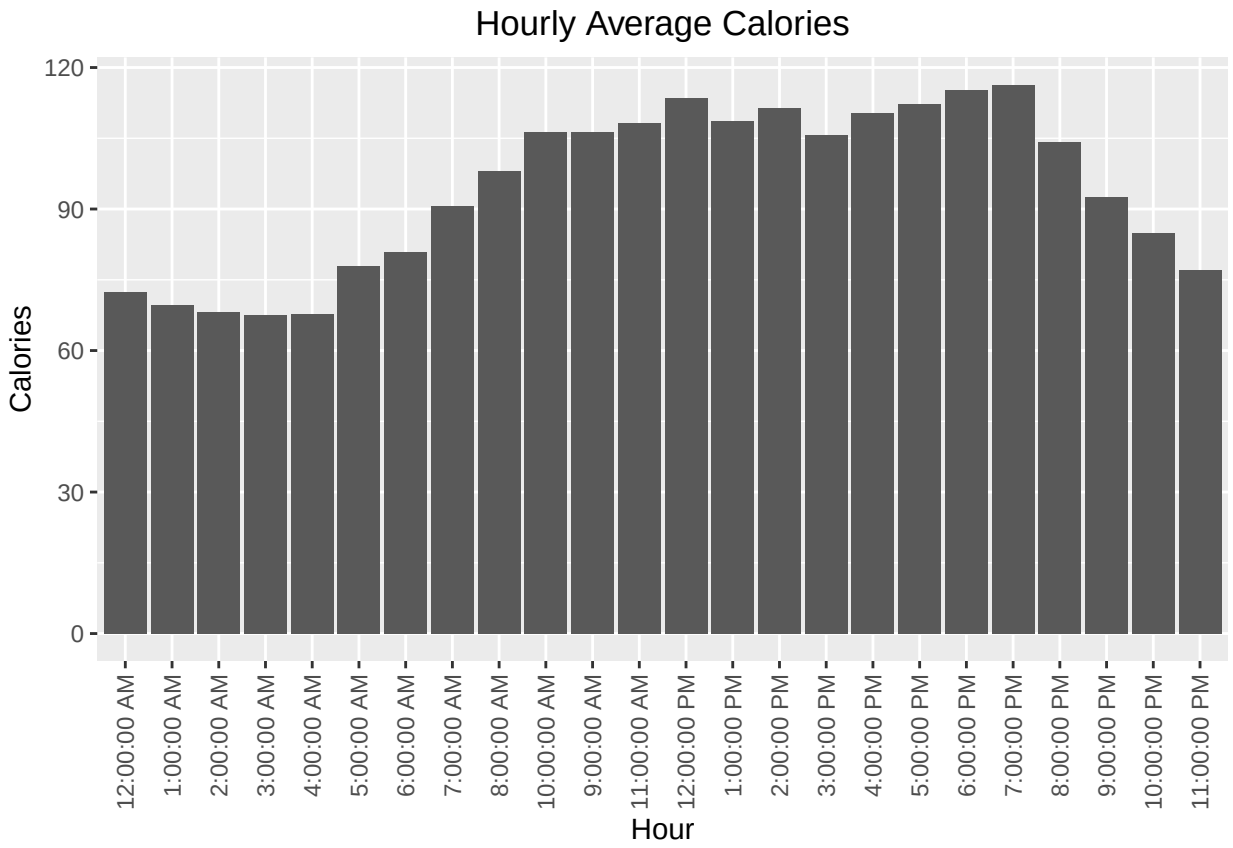
```
## # A tibble: 24 x 9
##    Time        avg_calories max_calories avg_steps max_steps avg_total_intensity
##    <chr>              <dbl>        <dbl>     <dbl>     <dbl>               <dbl>
##  1 12:00:00 AM         72.4          324     44.4       2854               2.41
##  2 1:00:00 AM          69.6          231     20.5        809               1.32
##  3 2:00:00 AM          68.1          149     10.6        586               0.760
##  4 3:00:00 AM          67.6          221      7.31       910               0.547
##  5 4:00:00 AM          67.8          175      9.76       723               0.578
##  6 5:00:00 AM          78.0          669     26.1       2780               3.84
##  7 6:00:00 AM          80.8          488    120.        3684               5.54
##  8 7:00:00 AM          90.6          489    267.        5660               9.61
##  9 8:00:00 AM          98.0          508    366.        6114              12.7
## 10 10:00:00 AM        106.           538    437.        6163              15.9
## # i 14 more rows
## # i 3 more variables: max_total_intensity <dbl>, avg_avg_intensity <dbl>,
## #   index <dbl>
```

Now let's see what the graphs of the averages over the course of a day look like.

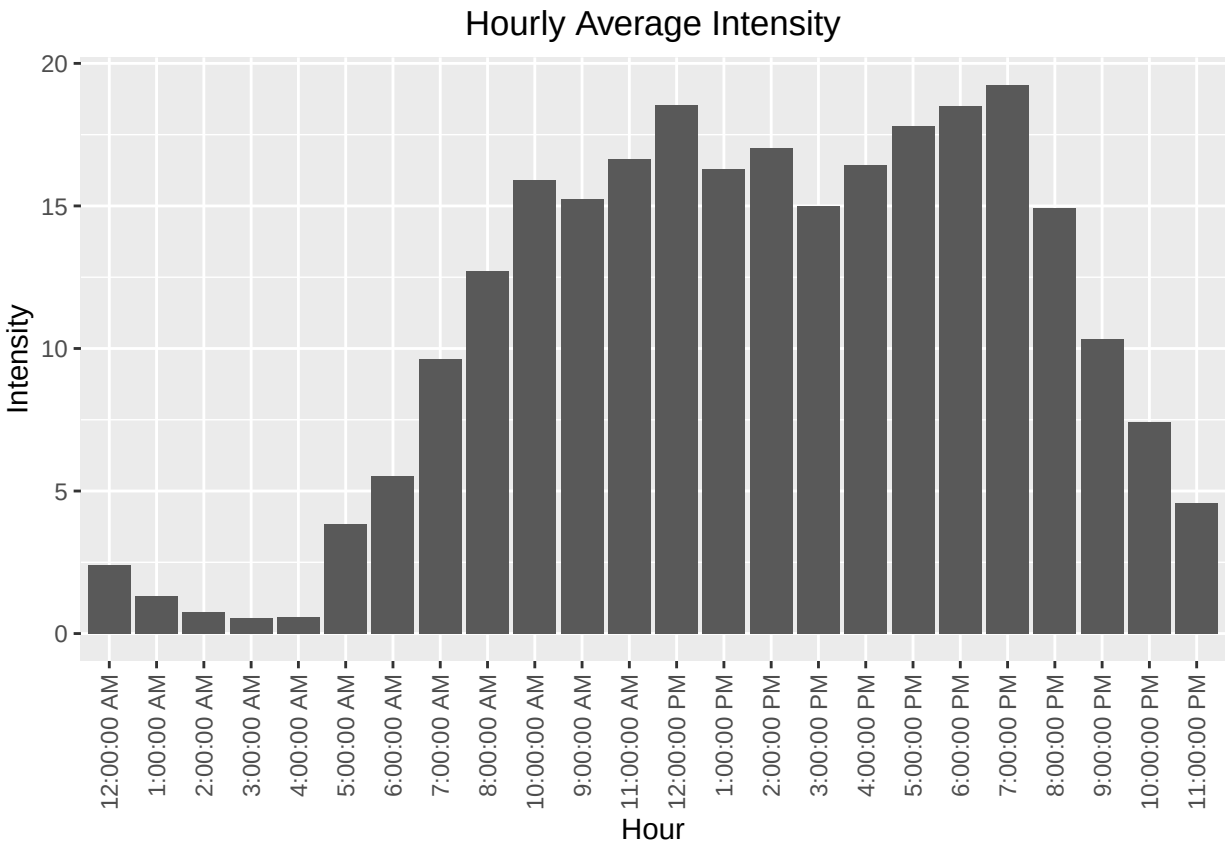
```
ggplot(sorted_hour_based_grouping, aes(x=reorder(Time,index), y=avg_steps, fill=avg_steps)) +
  geom_bar(stat="identity") +
  ggtitle("Hourly Average Steps") + xlab("Hour") + ylab("Steps") +
  theme(plot.title = element_text(hjust = 0.5)) +
  scale_x_discrete(guide = guide_axis(angle = 90))
```




```
ggplot(sorted_hour_based_grouping, aes(x=reorder(Time,index), y=avg_calories)) +
  geom_bar(stat="identity") +
  ggtitle("Hourly Average Calories") + xlab("Hour") + ylab("Calories") +
  theme(plot.title = element_text(hjust = 0.5)) +
  scale_x_discrete(guide = guide_axis(angle = 90))
```



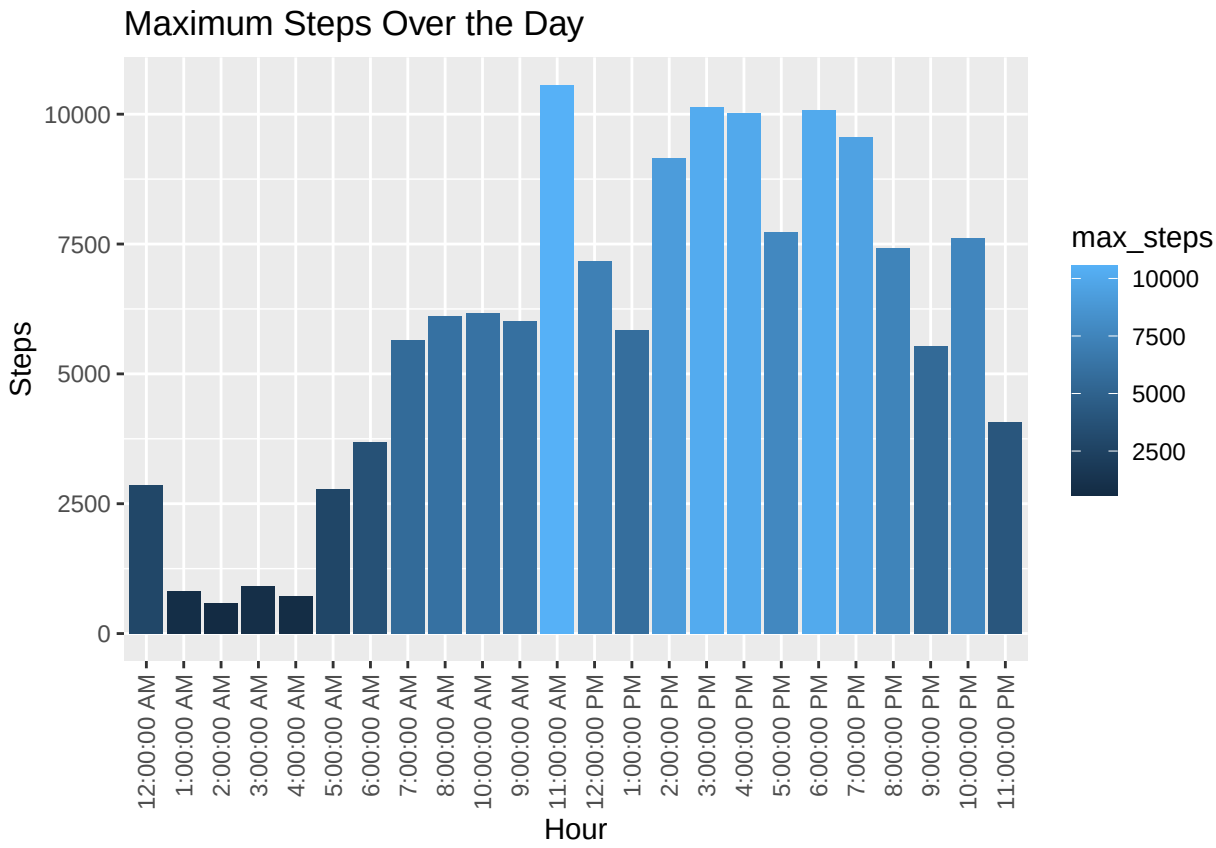

```
ggplot(sorted_hour_based_grouping, aes(x=reorder(Time,index), y=avg_total_intensity)) +
  geom_bar(stat="identity") +
  ggtitle("Hourly Average Intensity") + xlab("Hour") + ylab("Intensity") +
  theme(plot.title = element_text(hjust = 0.5))  +
  scale_x_discrete(guide = guide_axis(angle = 90))
```



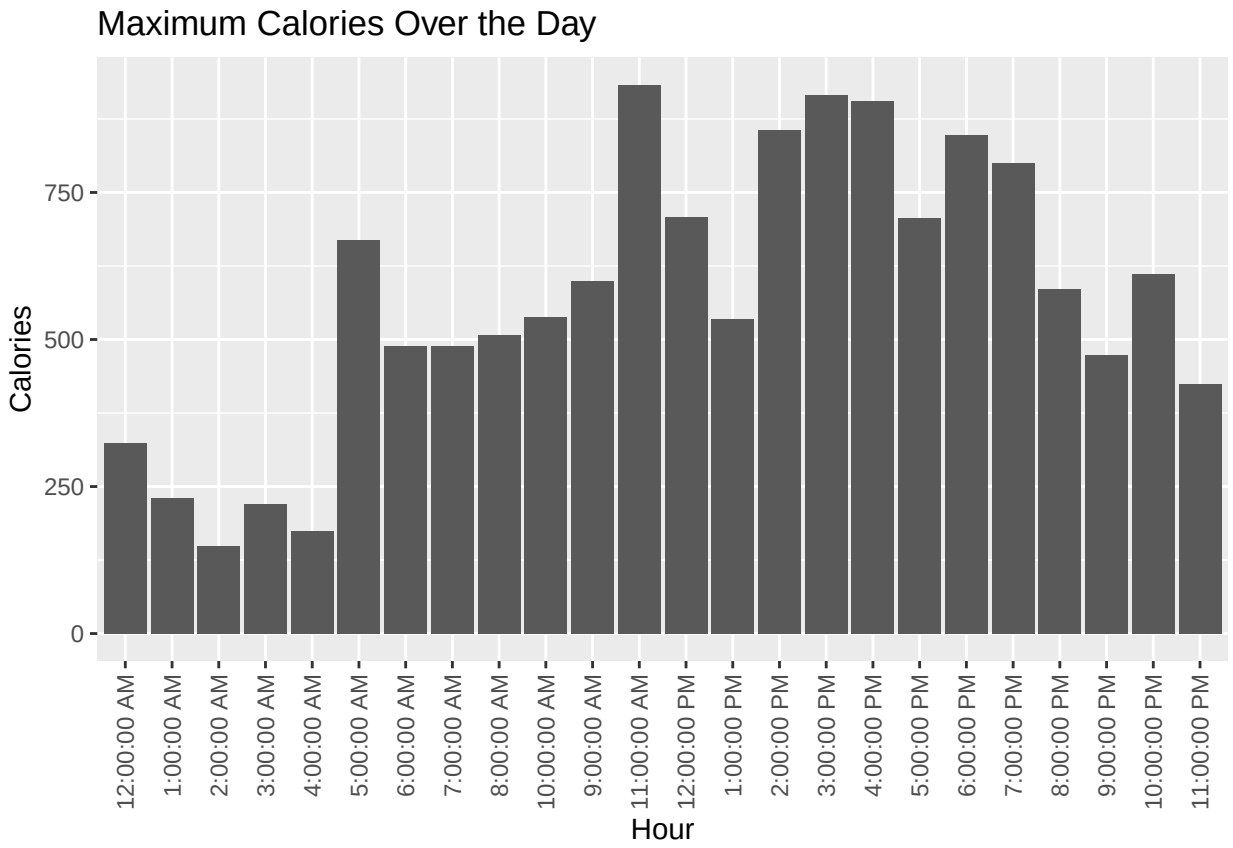

As you would expect, people are not very active in the middle of the night and are most active around the standard(ish) meal times - 12:00pm and 7:00pm.

Let's check the graphs for the maxes as well.

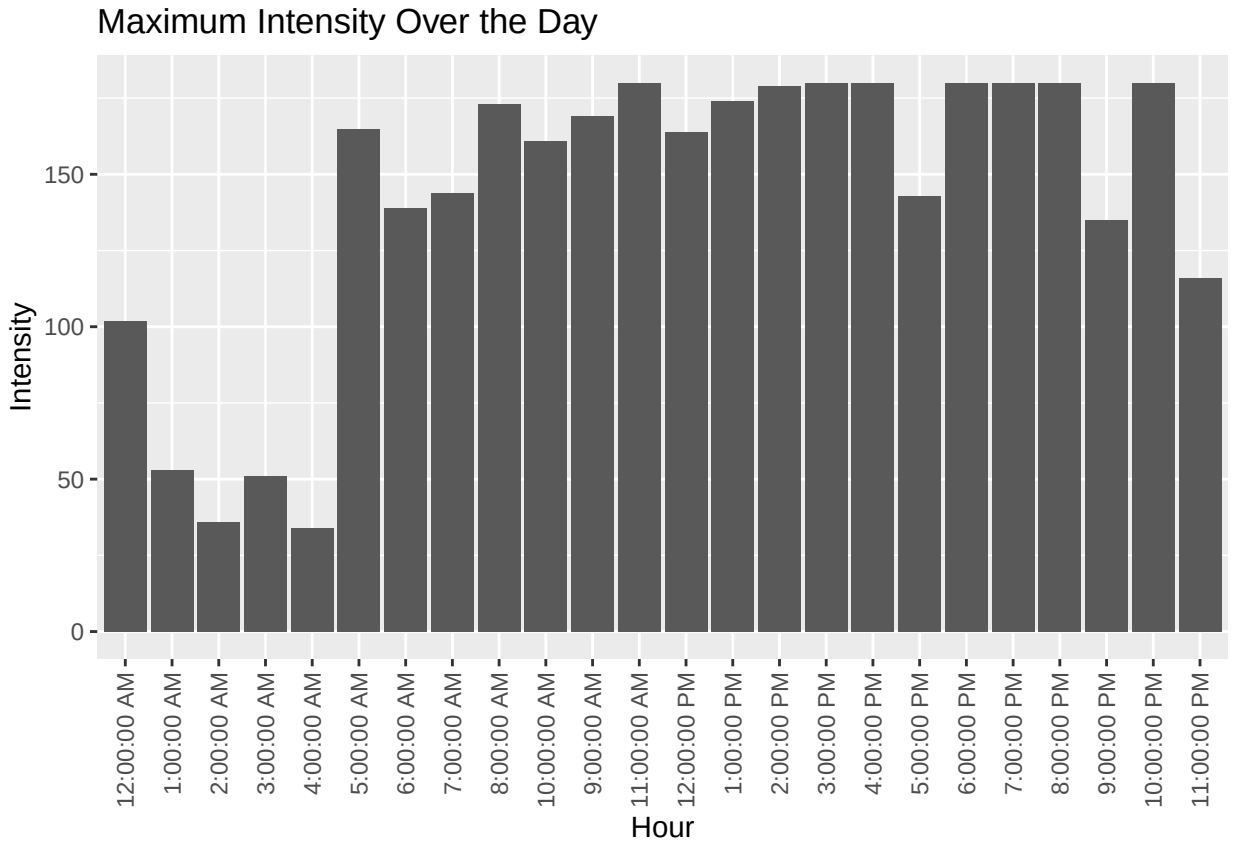
```
ggplot(sorted_hour_based_grouping, aes(x=reorder(Time,index), y=max_steps, fill=max_steps)) +
  geom_bar(stat="identity") +
  ggtitle("Maximum Steps Over the Day") + xlab("Hour") + ylab("Steps")  +
  scale_x_discrete(guide = guide_axis(angle = 90))
```




```
ggplot(sorted_hour_based_grouping, aes(x=reorder(Time,index), y=max_calories)) +
  geom_bar(stat="identity") +
  ggtitle("Maximum Calories Over the Day") + xlab("Hour") + ylab("Calories")  +
  scale_x_discrete(guide = guide_axis(angle = 90))
```




```
ggplot(sorted_hour_based_grouping, aes(x=reorder(Time,index), y=max_total_intensity)) +
  geom_bar(stat="identity") +
  ggtitle("Maximum Intensity Over the Day") + xlab("Hour") + ylab("Intensity")  +
  scale_x_discrete(guide = guide_axis(angle = 90))
```




The maximums generally follow the same trend, although the max intensities are more evenly spread with a notable and expected dip during the early morning/late night hours.

**1b. - Are people becoming more active over time as they track their activity?**

To answer this question, let's take a look at the maxima and averages of amount of time spent at each level of activity and the steps, calories, and distance each day and graph them. First we'll create a new dataframe with the summarized numbers.

```
day_based_grouping <- daily_activity %>%
  group_by(ActivityDate) %>%
  summarize(avg_sed_time = mean(SedentaryMinutes), max_sed_time = max(SedentaryMinutes),
            avg_la_time = mean(LightlyActiveMinutes), max_la_time = max(LightlyActiveMinutes),
            avg_fa_time = mean(FairlyActiveMinutes), max_fa_time = max(FairlyActiveMinutes),
            avg_va_time = mean(VeryActiveMinutes), max_va_time = max(VeryActiveMinutes),
            avg_calories = mean(Calories), max_calories = max(Calories),
            avg_steps = mean(TotalSteps), max_steps = max(TotalSteps),
            avg_total_distance = mean(TotalDistance), max_total_distance = max(TotalDistance)) %>%
  mutate(ActivityDate=as.Date(ActivityDate, format = "%m/%d/%Y")) %>%
  arrange(ActivityDate)

print(day_based_grouping)
```

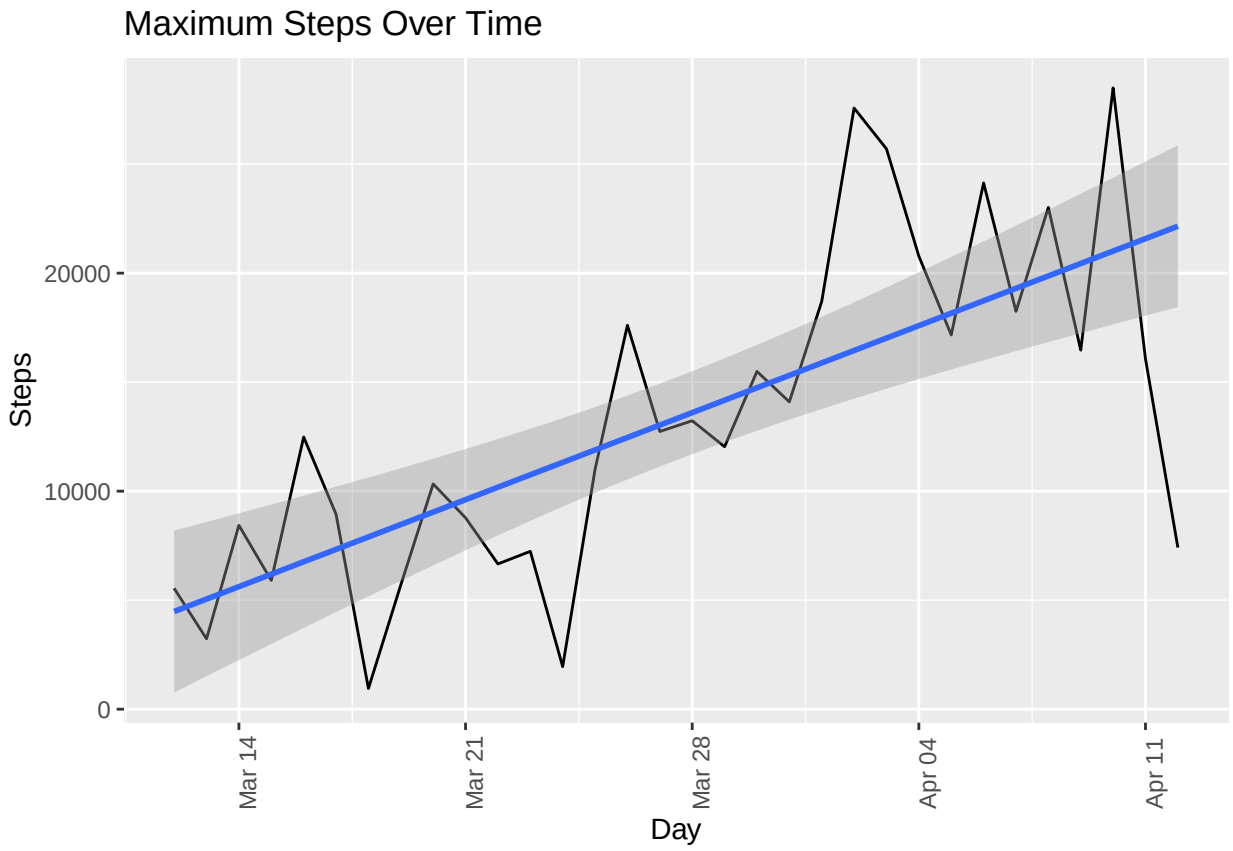
```
## # A tibble: 32 x 15
##    ActivityDate avg_sed_time max_sed_time avg_la_time max_la_time avg_fa_time
##    <date>              <dbl>        <dbl>       <dbl>       <dbl>       <dbl>
##  1 2016-03-12          1098.         1440       127           254         0
##  2 2016-03-13          1106.         1440        68           136         0
##  3 2016-03-14          1136.         1268       140           145         3.5
```

```
##  4 2016-03-15          1157          1440       108.          215          5
##  5 2016-03-16           998          1397       162           309         21
##  6 2016-03-17          1213          1440        23.5          47          0
##  7 2016-03-18          1423          1440        17            34          0
##  8 2016-03-19          1440          1440         0             0          0
##  9 2016-03-20          1439          1440         1             2          0
## 10 2016-03-21          1417          1440        23            46          0
## # i 22 more rows
## # i 9 more variables: max_fa_time <dbl>, avg_va_time <dbl>, max_va_time <dbl>,
## #   avg_calories <dbl>, max_calories <dbl>, avg_steps <dbl>, max_steps <dbl>,
## #   avg_total_distance <dbl>, max_total_distance <dbl>
```

Now we'll look at the max and average steps, calories, and distance.

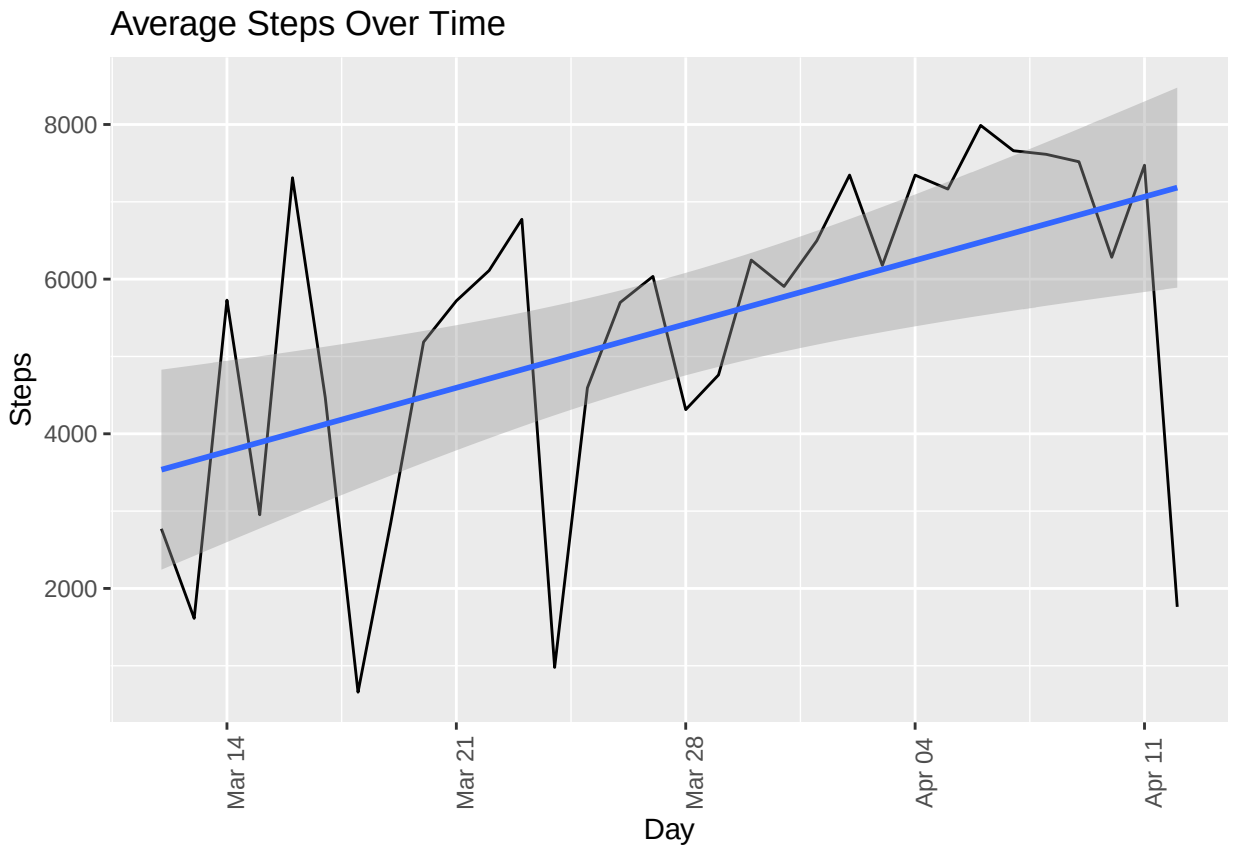
```
ggplot(day_based_grouping, aes(x=ActivityDate, y=max_steps, group = 1)) +
  geom_line() +
  ggtitle("Maximum Steps Over Time") + xlab("Day") + ylab("Steps")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



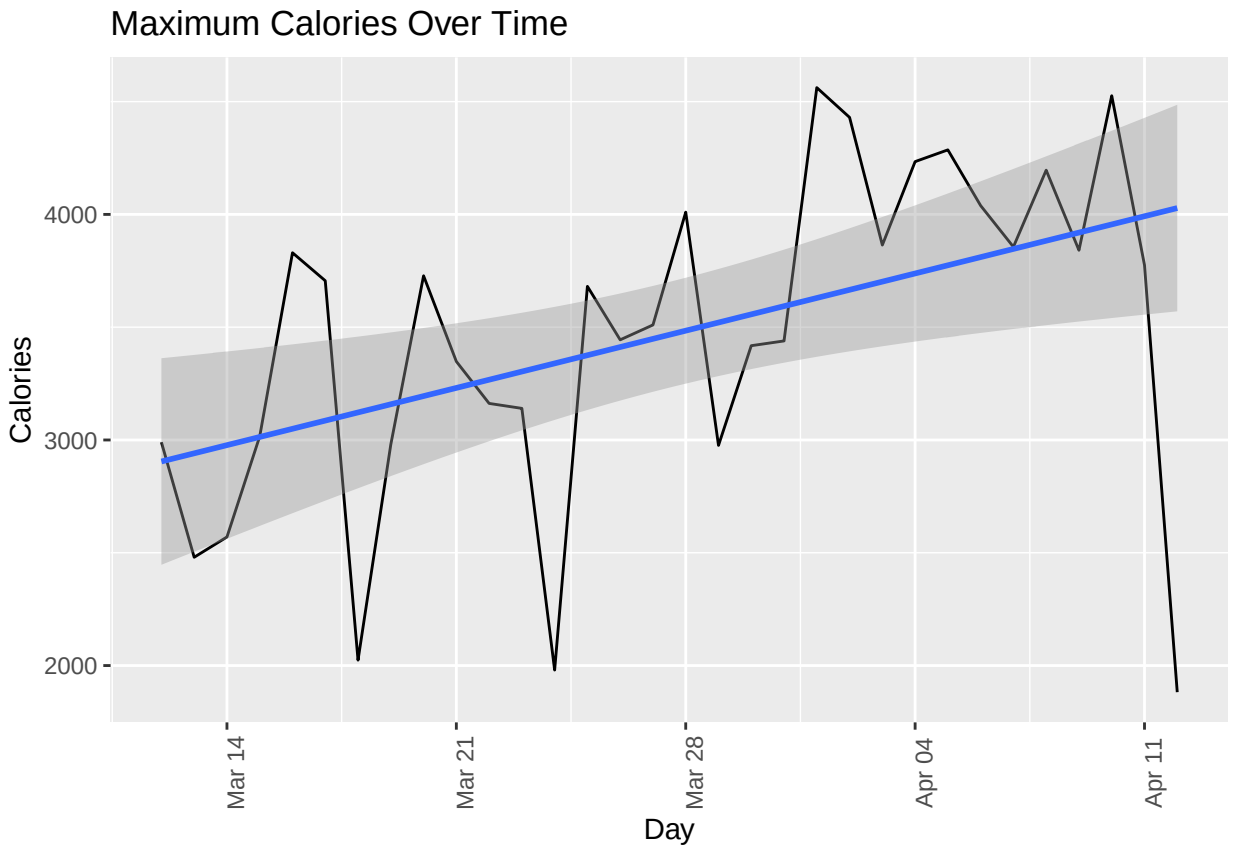

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=avg_steps, group = 1)) +
  geom_line() +
  ggtitle("Average Steps Over Time") + xlab("Day") + ylab("Steps")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



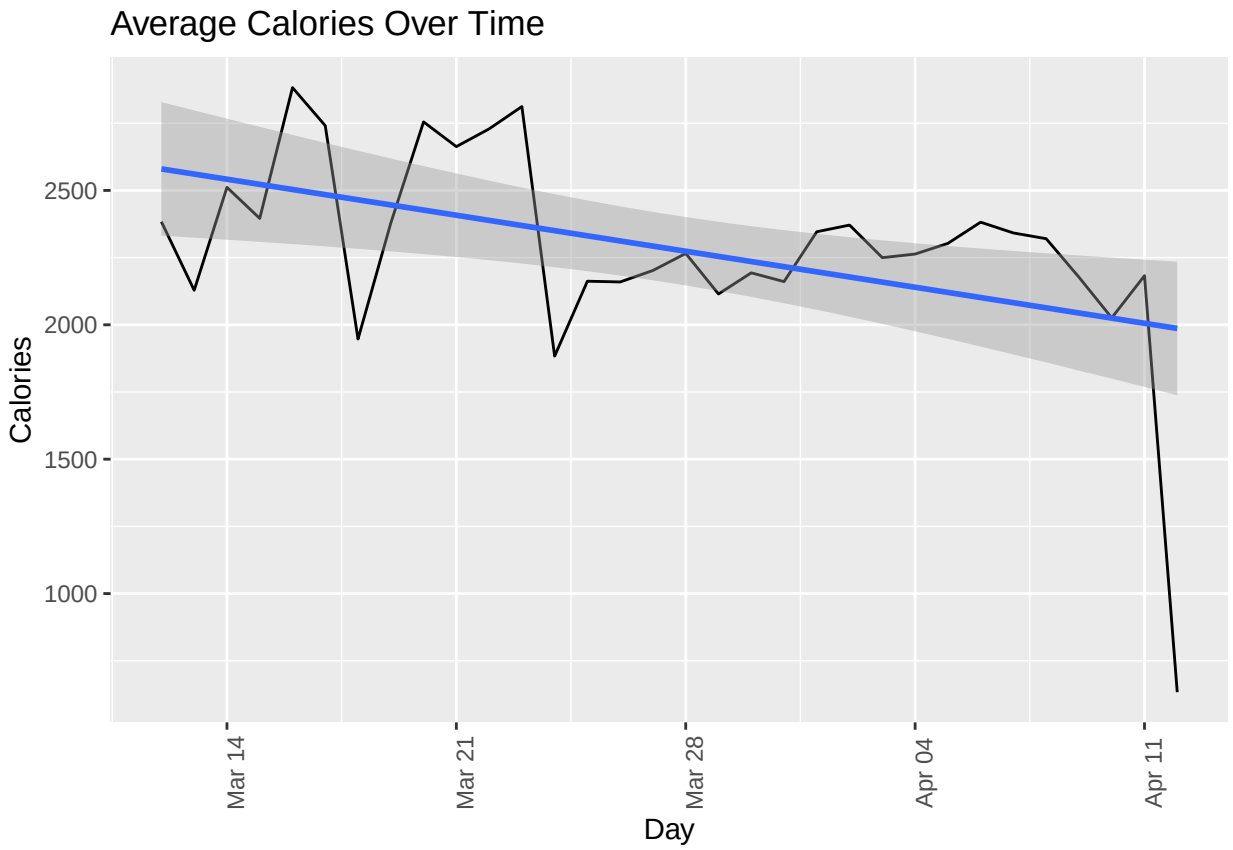

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=max_calories, group = 1)) +
  geom_line() +
  ggtitle("Maximum Calories Over Time") + xlab("Day") + ylab("Calories")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



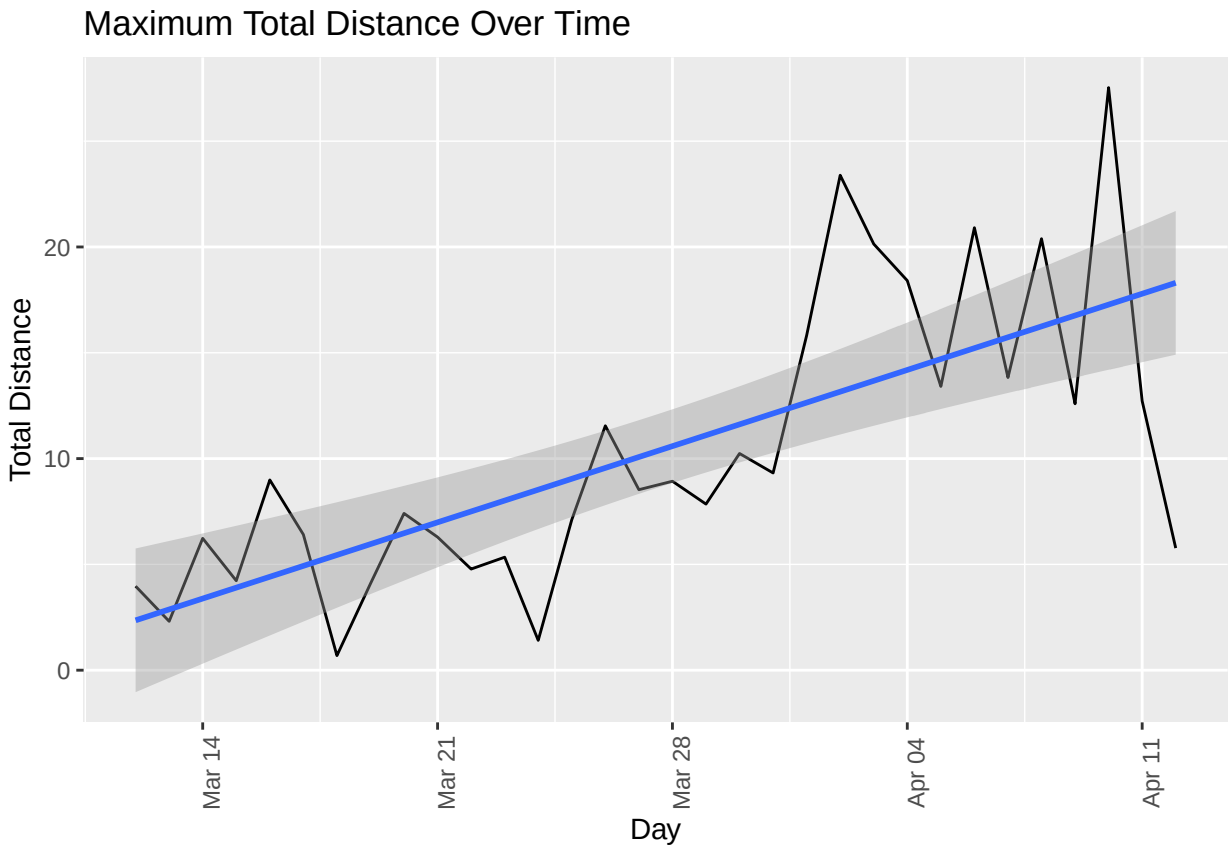

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=avg_calories, group = 1)) +
  geom_line() +
  ggtitle("Average Calories Over Time") + xlab("Day") + ylab("Calories")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



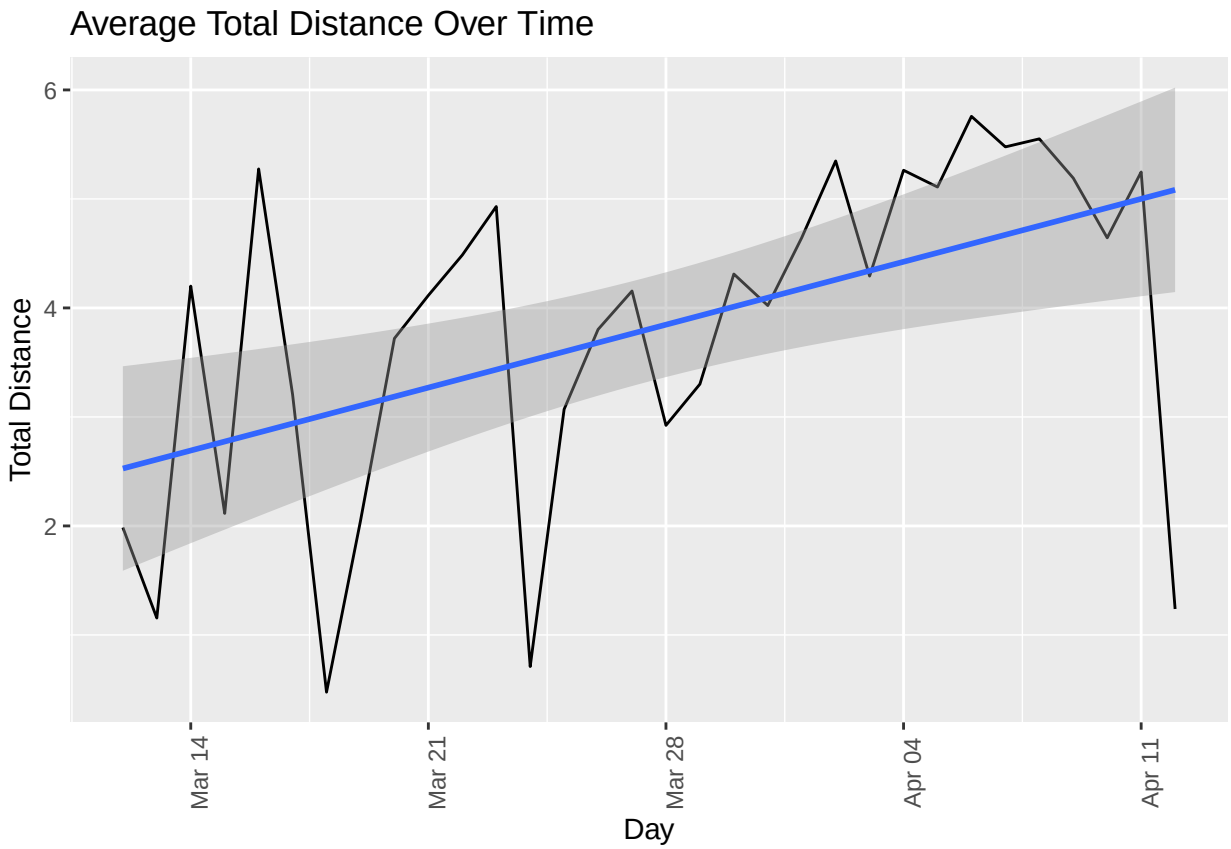

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=max_total_distance, group = 1)) +
  geom_line() +
  ggtitle("Maximum Total Distance Over Time") + xlab("Day") + ylab("Total Distance")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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




```
ggplot(day_based_grouping, aes(x=ActivityDate, y=avg_total_distance, group = 1)) +
  geom_line() +
  ggtitle("Average Total Distance Over Time") + xlab("Day") + ylab("Total Distance")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

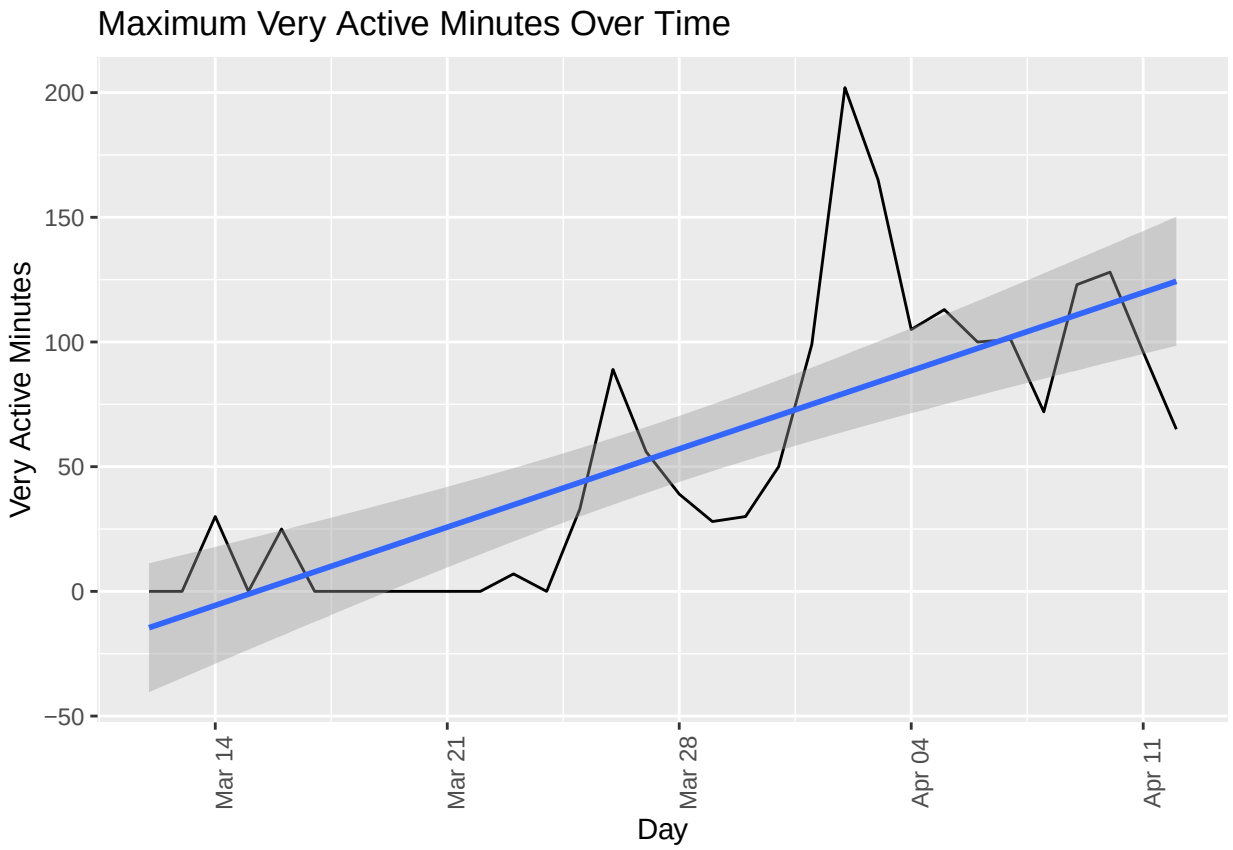
```
## `geom_smooth()` using formula = 'y ~ x'
```




And then look at the times spent at the four activity levels.

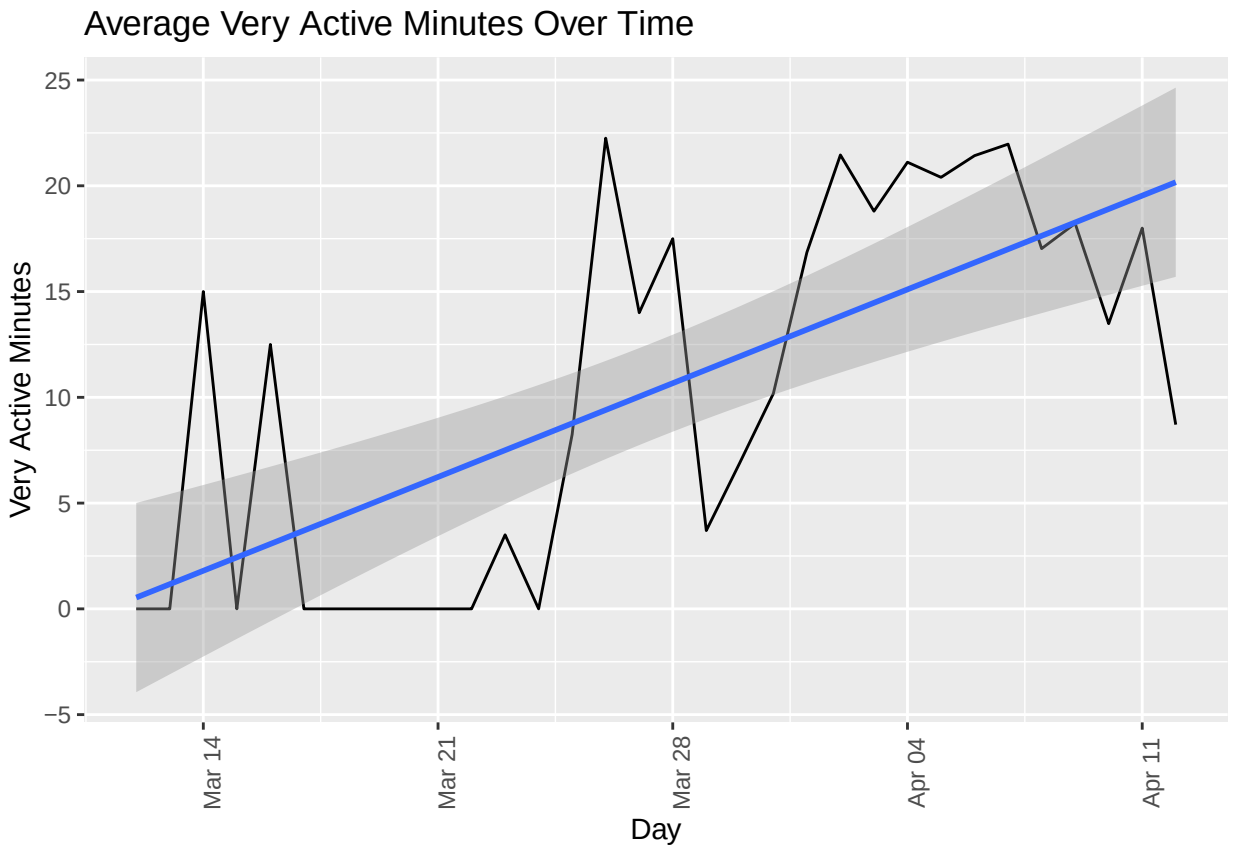
```
ggplot(day_based_grouping, aes(x=ActivityDate, y=max_va_time, group = 1)) +
  geom_line() +
  ggtitle("Maximum Very Active Minutes Over Time") + xlab("Day") + ylab("Very Active Minutes")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



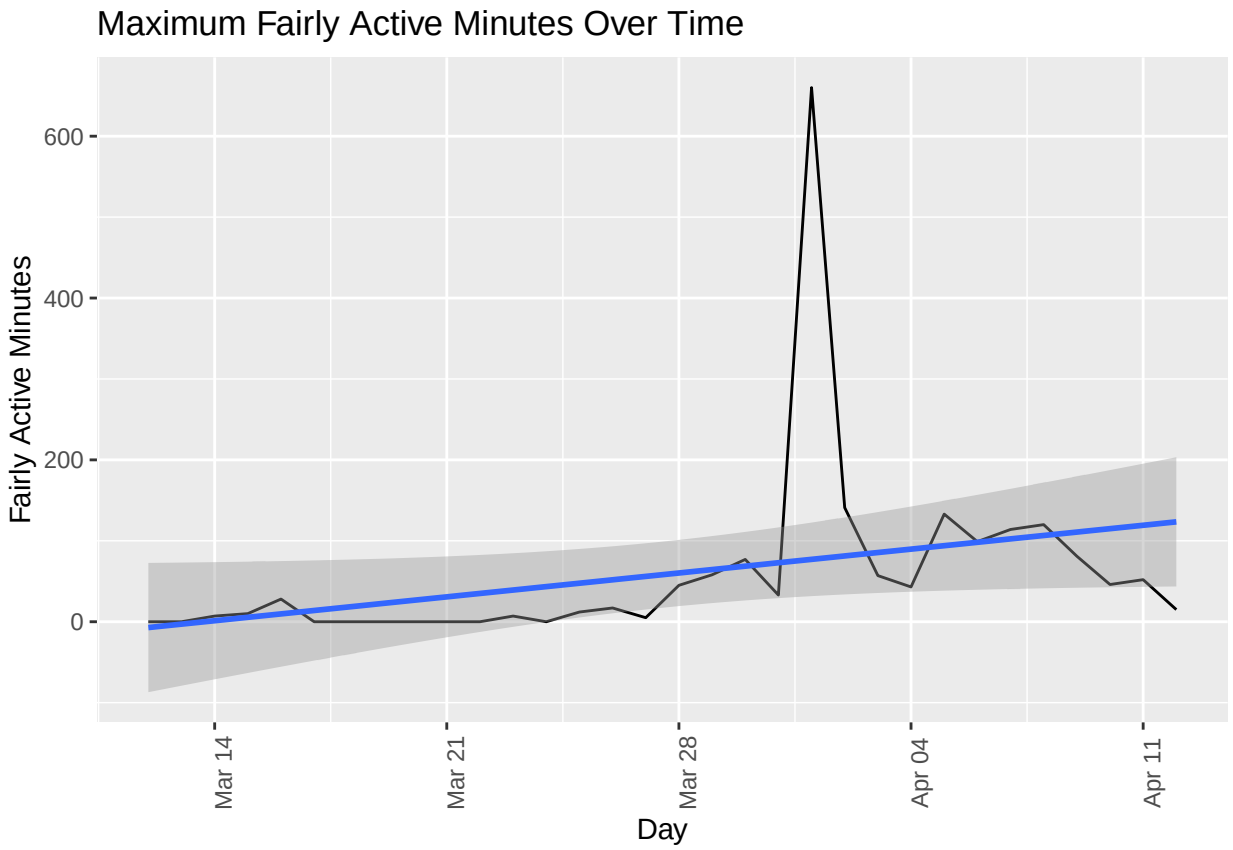

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=avg_va_time, group = 1)) +
  geom_line() +
  ggtitle("Average Very Active Minutes Over Time") + xlab("Day") + ylab("Very Active Minutes")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



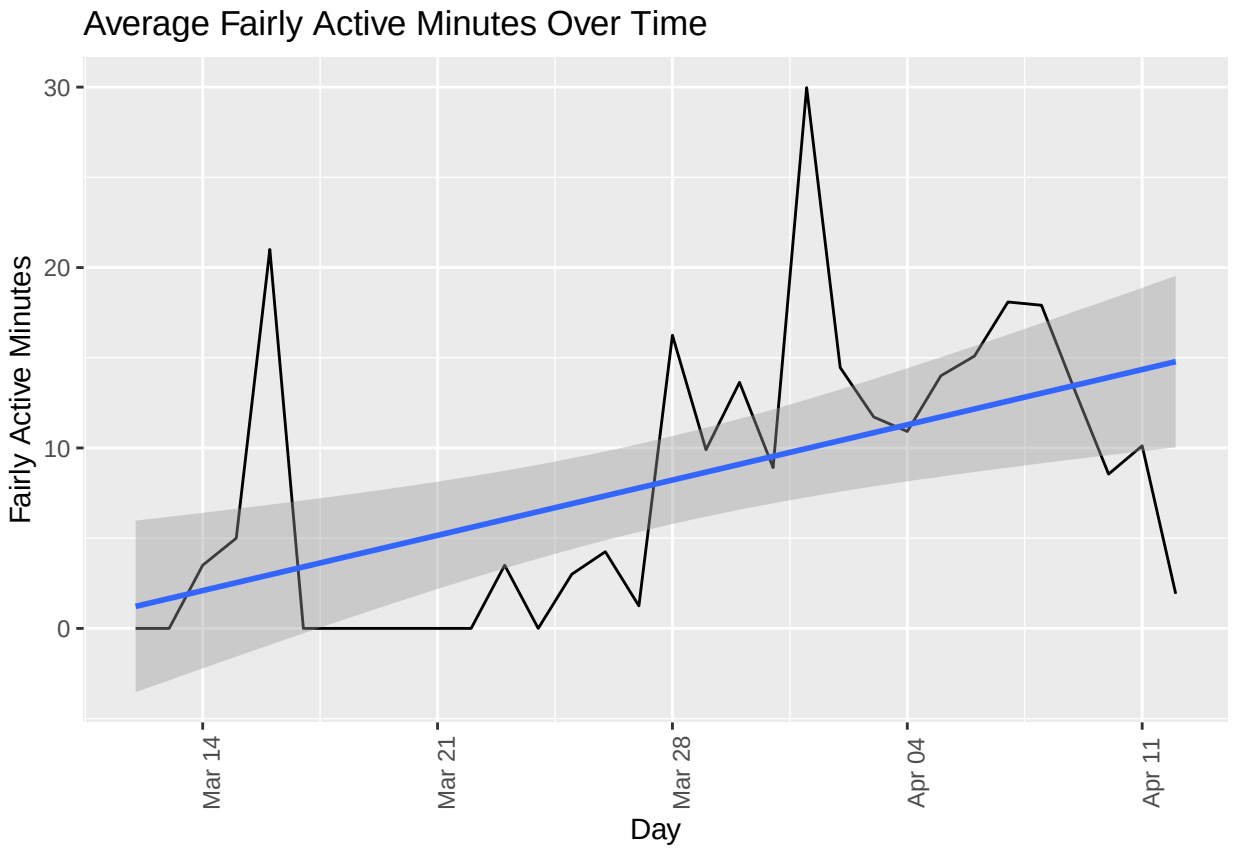

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=max_fa_time, group = 1)) +
  geom_line() +
  ggtitle("Maximum Fairly Active Minutes Over Time") + xlab("Day") + ylab("Fairly Active Minutes")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



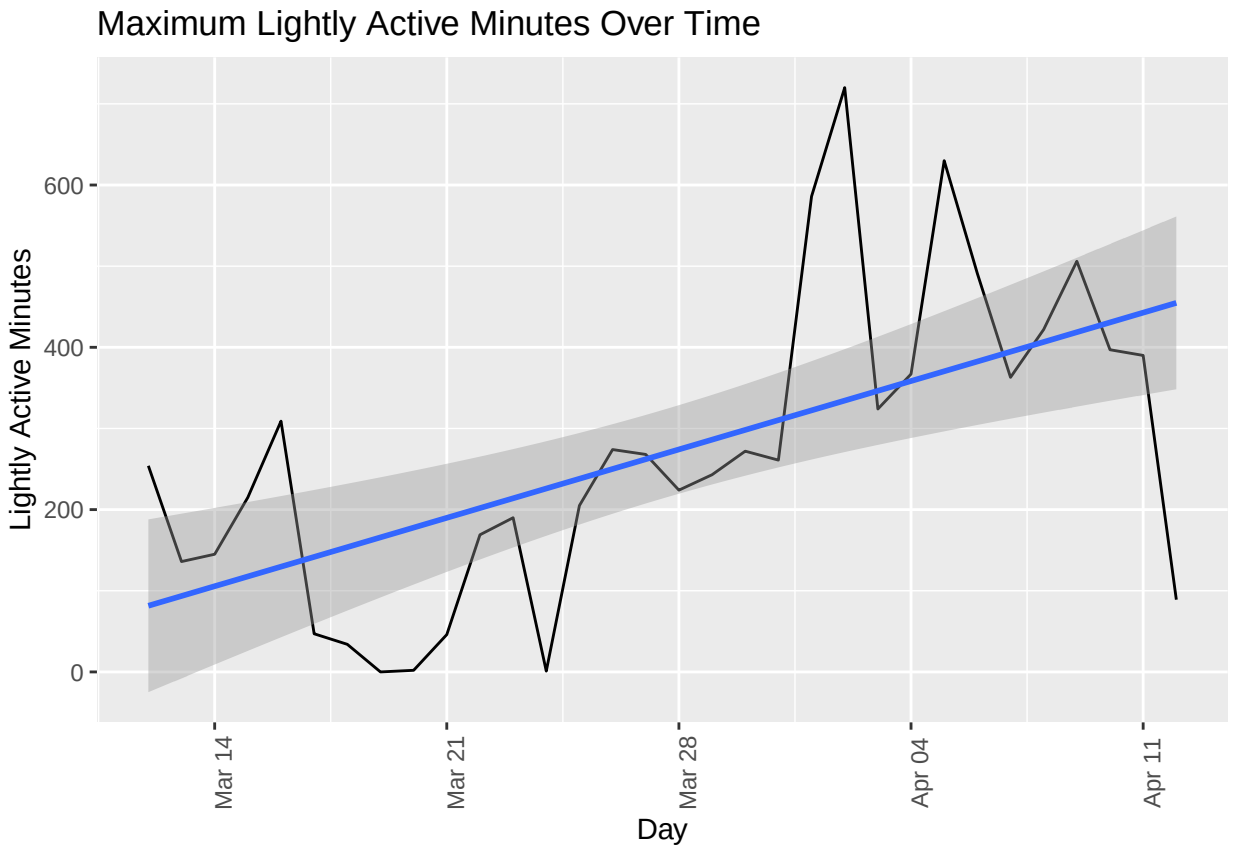

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=avg_fa_time, group = 1)) +
  geom_line() +
  ggtitle("Average Fairly Active Minutes Over Time") + xlab("Day") + ylab("Fairly Active Minutes")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



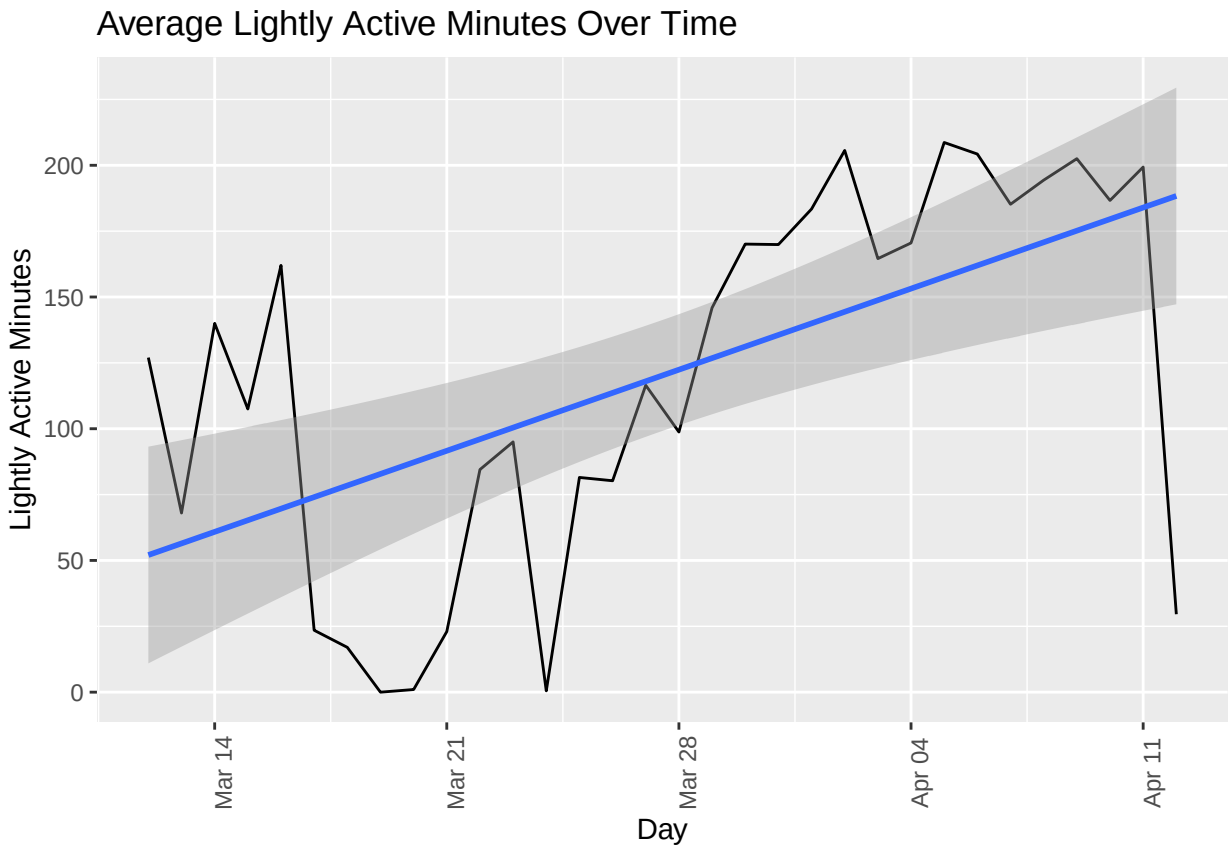

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=max_la_time, group = 1)) +
  geom_line() +
  ggtitle("Maximum Lightly Active Minutes Over Time") + xlab("Day") + ylab("Lightly Active Minutes")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



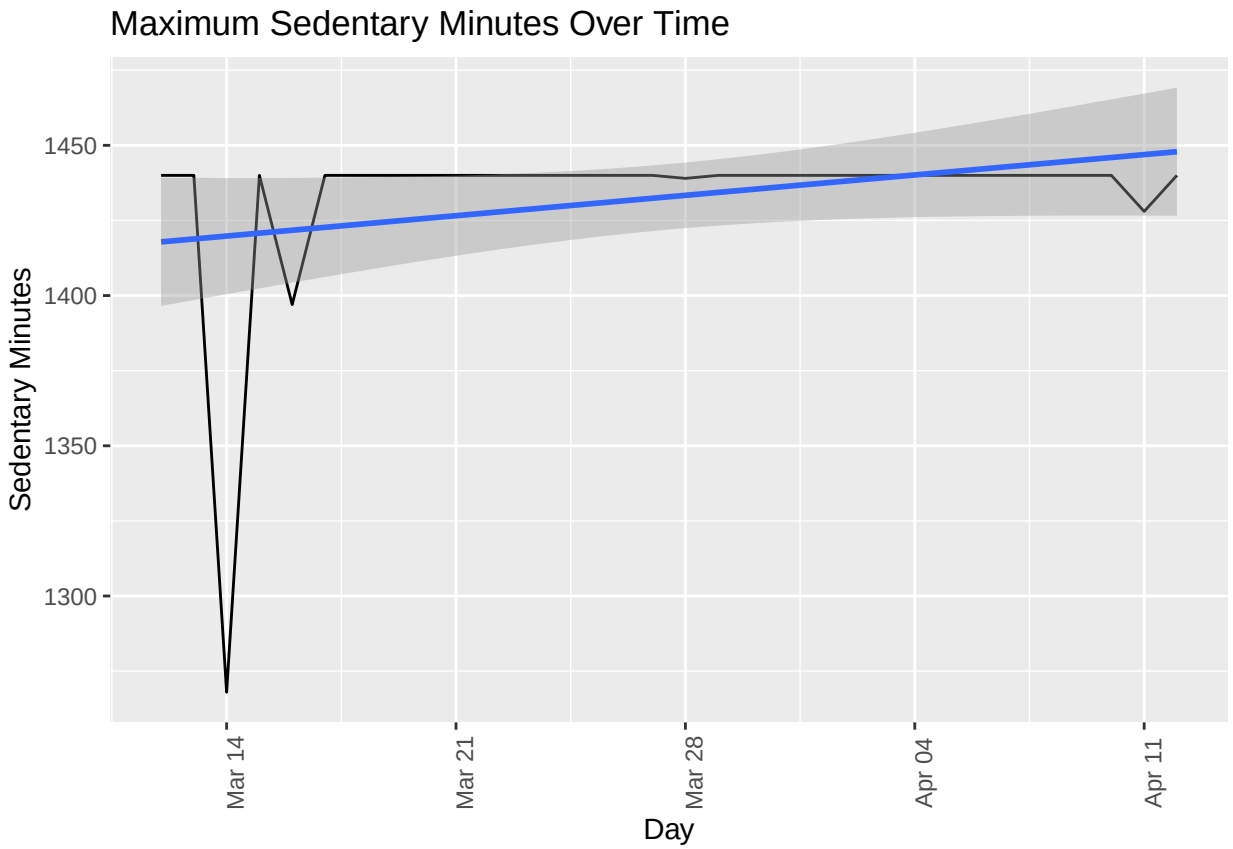

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=avg_la_time, group = 1)) +
  geom_line() +
  ggtitle("Average Lightly Active Minutes Over Time") + xlab("Day") + ylab("Lightly Active Minutes")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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



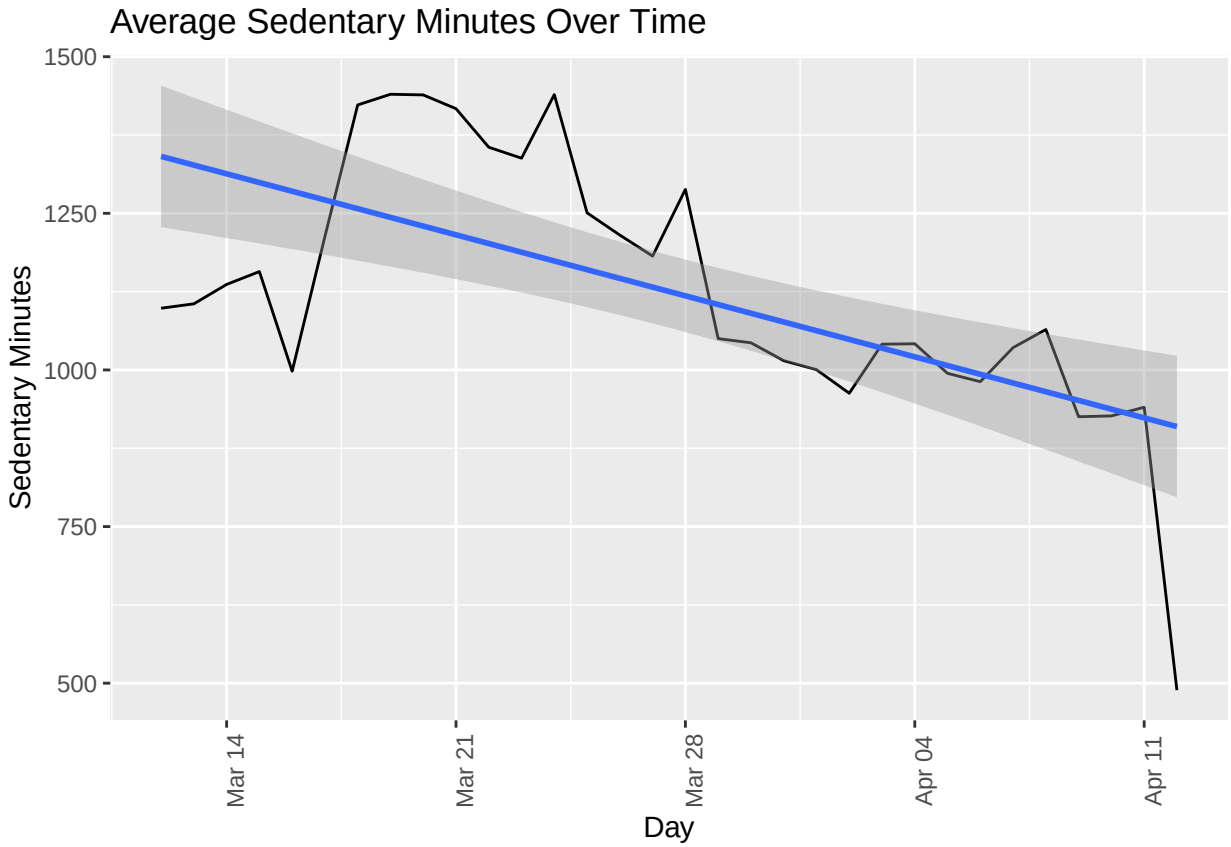

```
ggplot(day_based_grouping, aes(x=ActivityDate, y=max_sed_time, group = 1)) +
  geom_line() +
  ggtitle("Maximum Sedentary Minutes Over Time") + xlab("Day") + ylab("Sedentary Minutes")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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




```
ggplot(day_based_grouping, aes(x=ActivityDate, y=avg_sed_time, group = 1)) +
  geom_line() +
  ggtitle("Average Sedentary Minutes Over Time") + xlab("Day") + ylab("Sedentary Minutes")  +
  geom_smooth(method = "lm") +
  theme(axis.text.x = element_text(angle = 90))
```

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




The data strongly support the idea that smart device users are increasing their activity over the time during which they are using said smart devices. The only metric looked at that did not increase during this period is average calories, which would require a deeper look into that data for more context. The average and maximum very active minutes both show a marked increase over the data period and both graphs for the fairly and lightly active minutes show an increase, albeit not as steep of one (with the possible exception of average lightly active minutes). While the maximum sedentary minutes graph shows a slight upward trend, the average sedentary minutes graph shows a notable decrease over the data period. There is a sharp decrease for 4/12 in several of the graphs that will be explored shortly.

### 2. How much are people walking, when, and at what intensities?

This is answered by the graphs created for 1a. Average steps are highest at 12:00pm, 6:00pm, and 7:00pm with over 500 steps on average for all three hours. The average intensities for these times are 18.5431727, 18.5206445, and 19.2477341 respectively.

### 3b. Do the reports for the same people show any interesting trends?

The two users with more than two reports are 6962181067 and 8877689391, so let's take a look at their entries, starting with 1067.

```
user1_reports <- weightlog_info[weightlog_info$Id==6962181067,]
print(user1_reports)
```

```
## # A tibble: 14 x 9
##          Id Date  Time  WeightKg WeightPounds   Fat   BMI IsManualReport   LogId
##       <dbl> <chr> <chr>    <dbl>        <dbl> <dbl> <dbl> <lgl>            <dbl>
##  1   6.96e9 3/30~ 11:5~     61.5         136.    NA  24.0 TRUE           1.46e12
##  2   6.96e9 3/31~ 11:5~     61.5         136.    NA  24.0 TRUE           1.46e12
##  3   6.96e9 4/1/~ 11:5~     60.9         134.    NA  23.8 TRUE           1.46e12
```

```
##  4   6.96e9 4/2/~ 11:5~     61.2         135.    NA  23.9 TRUE           1.46e12
##  5   6.96e9 4/3/~ 11:5~     61.5         136.    NA  24.0 TRUE           1.46e12
##  6   6.96e9 4/4/~ 11:5~     62.4         138.    NA  24.4 TRUE           1.46e12
##  7   6.96e9 4/5/~ 11:5~     61.7         136.    NA  24.1 TRUE           1.46e12
##  8   6.96e9 4/6/~ 11:5~     62.2         137.    NA  24.3 TRUE           1.46e12
##  9   6.96e9 4/7/~ 11:5~     62.2         137.    NA  24.3 TRUE           1.46e12
## 10   6.96e9 4/8/~ 11:5~     61.7         136.    NA  24.1 TRUE           1.46e12
## 11   6.96e9 4/9/~ 11:5~     62.1         137.    NA  24.2 TRUE           1.46e12
## 12   6.96e9 4/10~ 11:5~     62.5         138.    NA  24.4 TRUE           1.46e12
## 13   6.96e9 4/11~ 11:5~     62.2         137.    NA  24.3 TRUE           1.46e12
## 14   6.96e9 4/12~ 11:5~     62.5         138.    NA  24.4 TRUE           1.46e12
```

```
user1_activity <- combined_hourly[combined_hourly$Id==6962181067,] %>%
  group_by(Date) %>%
  summarize(avg_calories = mean(Calories), total_calories = sum(Calories), avg_steps = mean(StepTotal),
  mutate(Date=as.Date(Date, format = "%m/%d/%Y")) %>%
  arrange(Date)
print(user1_activity)
```

```
## # A tibble: 32 x 8
##    Date       avg_calories total_calories avg_steps total_steps
##    <date>            <dbl>          <dbl>     <dbl>       <dbl>
##  1 2016-03-12         94.8           2275      695.       16671
##  2 2016-03-13         85.4           2049      426.       10236
##  3 2016-03-14         94.1           2259      583.       13999
##  4 2016-03-15         83.8           2010      423.       10156
##  5 2016-03-16        105.            2523      676.       16236
##  6 2016-03-17        117.            2803     1088.       26100
##  7 2016-03-18        120.            2876      961.       23071
##  8 2016-03-19         98.1           2354      851        20424
##  9 2016-03-20        102.            2445      836.       20053
## 10 2016-03-21        102             2448      847.       20329
## # i 22 more rows
## # i 3 more variables: avg_total_intensity <dbl>, max_total_intensity <dbl>,
## #   avg_avg_intensity <dbl>
```

And then 9391.

```
user2_reports <- weightlog_info[weightlog_info$Id==8877689391,]
print(user2_reports)
```

```
## # A tibble: 9 x 9
##          Id Date  Time  WeightKg WeightPounds   Fat   BMI IsManualReport   LogId
##       <dbl> <chr> <chr>    <dbl>        <dbl> <dbl> <dbl> <lgl>            <dbl>
## 1    8.88e9 4/1/~ 6:49~     85.5         188.    NA  25.6 FALSE          1.46e12
## 2    8.88e9 4/4/~ 6:53~     86.6         191.    NA  25.9 FALSE          1.46e12
## 3    8.88e9 4/5/~ 6:40~     86           190.    NA  25.8 FALSE          1.46e12
## 4    8.88e9 4/6/~ 6:49~     86.3         190.    NA  25.8 FALSE          1.46e12
## 5    8.88e9 4/7/~ 6:15~     85.1         188.    NA  25.5 FALSE          1.46e12
## 6    8.88e9 4/8/~ 6:39~     85           187.    NA  25.4 FALSE          1.46e12
## 7    8.88e9 4/9/~ 8:06~     85.4         188.    NA  25.6 FALSE          1.46e12
## 8    8.88e9 4/11~ 6:58~     86.1         190.    NA  25.8 FALSE          1.46e12
## 9    8.88e9 4/12~ 6:47~     85.8         189.    NA  25.7 FALSE          1.46e12
```

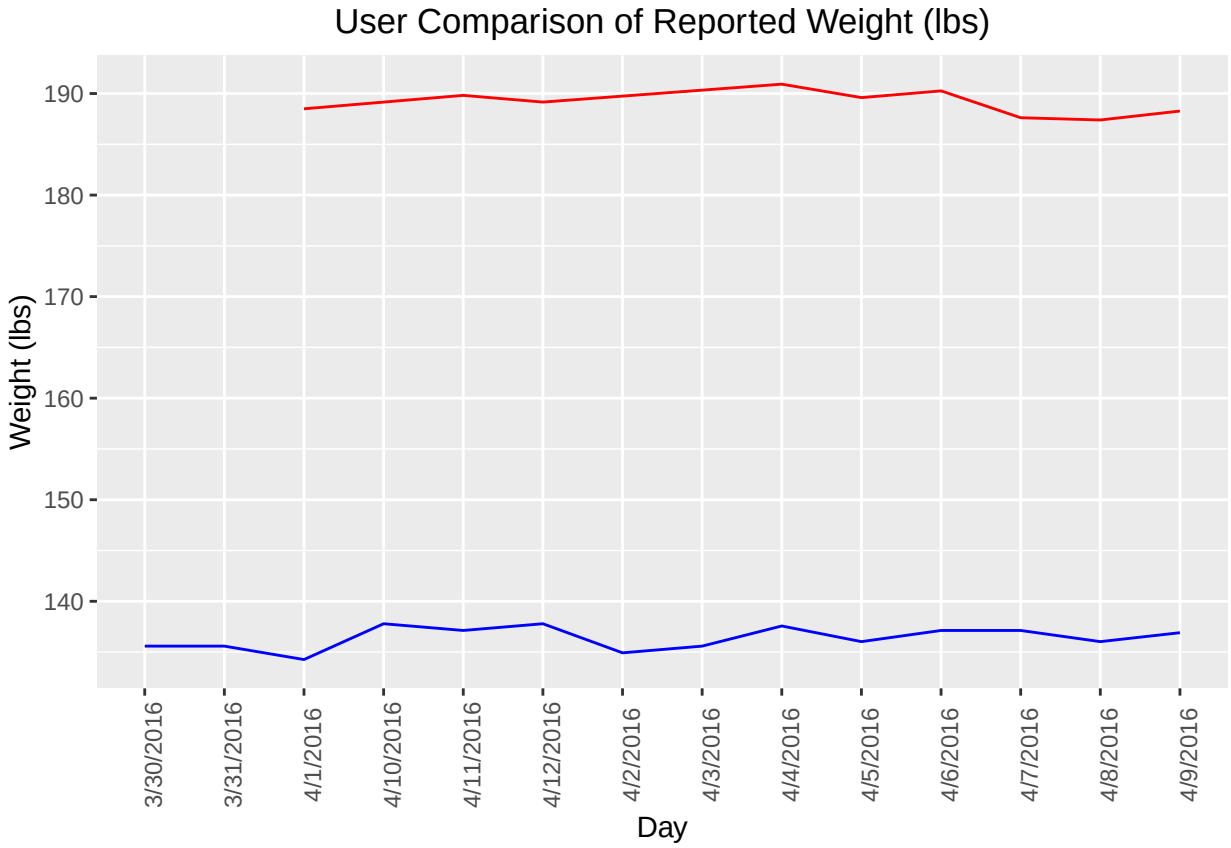
```
user2_activity <- combined_hourly[combined_hourly$Id==8877689391,] %>%
  group_by(Date) %>%
```

```
  summarize(avg_calories = mean(Calories), total_calories = sum(Calories), avg_steps = mean(StepTotal),
  mutate(Date=as.Date(Date, format = "%m/%d/%Y")) %>%
  arrange(Date)
print(user2_activity)
```

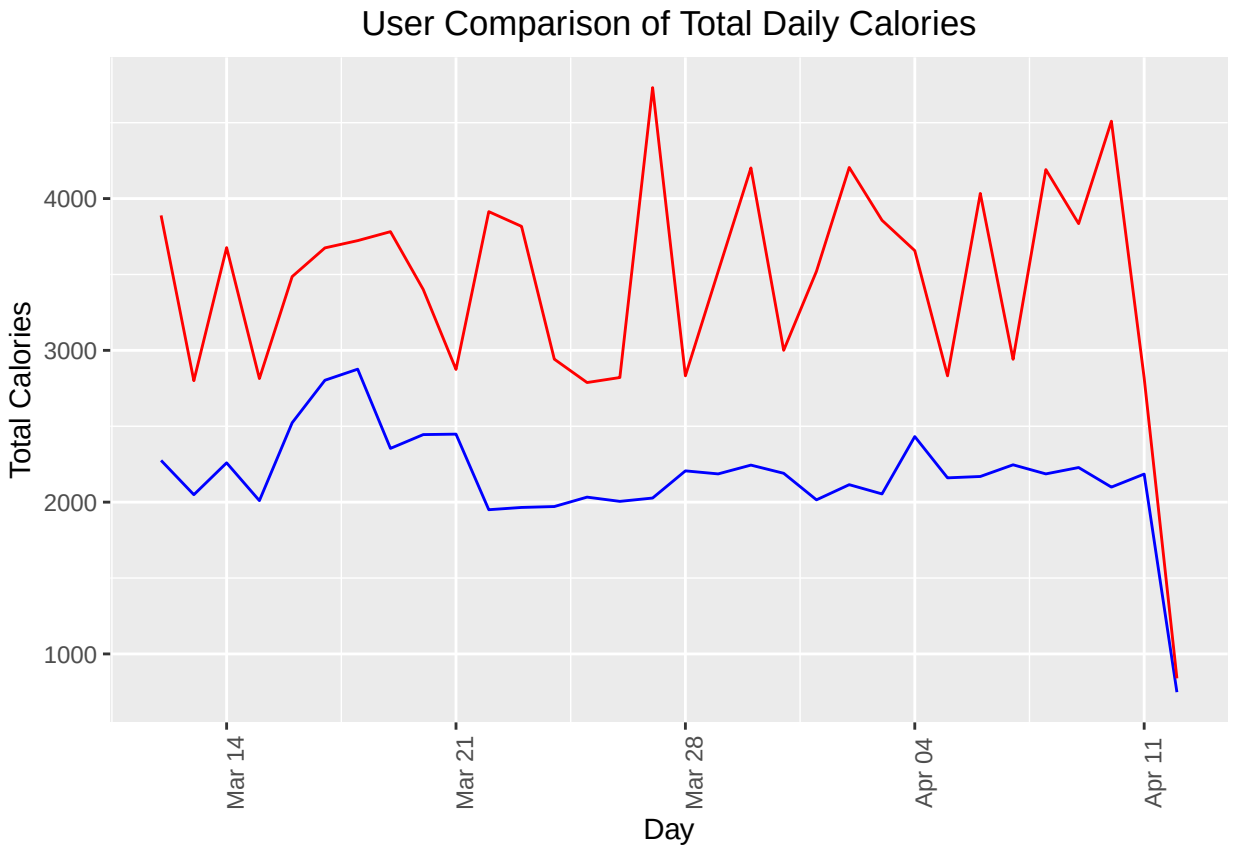
```
## # A tibble: 32 x 8
##    Date       avg_calories total_calories avg_steps total_steps
##    <date>            <dbl>          <dbl>     <dbl>       <dbl>
##  1 2016-03-12         162.           3889      810.       19442
##  2 2016-03-13         117.           2800      441.       10594
##  3 2016-03-14         153.           3677      835.       20039
##  4 2016-03-15         117.           2814      466.       11176
##  5 2016-03-16         145.           3486      748.       17963
##  6 2016-03-17         153.           3675      633.       15182
##  7 2016-03-18         155.           3723      894.       21467
##  8 2016-03-19         158.           3782      615.       14761
##  9 2016-03-20         142.           3401      724.       17377
## 10 2016-03-21         120.           2874      500.       11999
## # i 22 more rows
## # i 3 more variables: avg_total_intensity <dbl>, max_total_intensity <dbl>,
## #   avg_avg_intensity <dbl>
```

Now that we have those two isolated, let's take a look at some graphs comparing their activity.

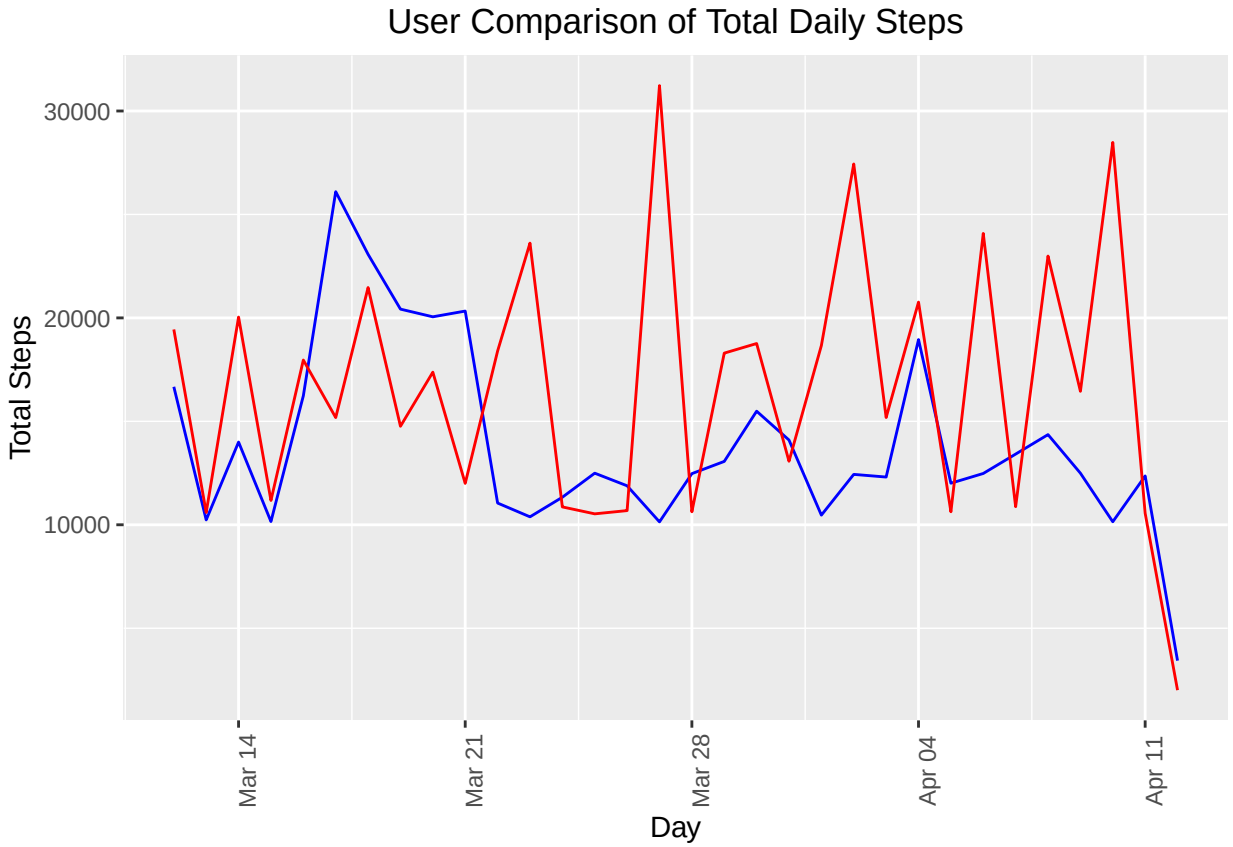
```
ggplot() +
  geom_line(data = user1_reports, aes(x = Date, y = WeightPounds, group = 1), color = "blue") +
  geom_line(data = user2_reports, aes(x = Date, y = WeightPounds, group = 1), color = "red") +
  ggtitle("User Comparison of Reported Weight (lbs)") + xlab("Day") + ylab("Weight (lbs)") +
  theme(plot.title = element_text(hjust = 0.5))  +
  theme(axis.text.x = element_text(angle = 90))
```




```
ggplot() +
  geom_line(data = user1_activity, aes(x = Date, y = total_calories, group = 1), color = "blue") +
  geom_line(data = user2_activity, aes(x = Date, y = total_calories, group = 1), color = "red") +
  ggtitle("User Comparison of Total Daily Calories") + xlab("Day") + ylab("Total Calories") +
  theme(plot.title = element_text(hjust = 0.5))  +
  theme(axis.text.x = element_text(angle = 90))
```




```
ggplot() +
  geom_line(data = user1_activity, aes(x = Date, y = total_steps, group = 1), color = "blue") +
  geom_line(data = user2_activity, aes(x = Date, y = total_steps, group = 1), color = "red") +
  ggtitle("User Comparison of Total Daily Steps") + xlab("Day") + ylab("Total Steps") +
  theme(plot.title = element_text(hjust = 0.5))  +
  theme(axis.text.x = element_text(angle = 90))
```




There isn't really enough information to find noteworthy trends for these two users. Both ended within a single pound of where they began - which is what you would probably expect given the reports span 2 weeks - and their steps and calories stayed roughly within the same range, albeit with higher maxes as time passed. I'm guessing the dip at the end is because the data for 4/12 doesn't represent the entire day. Let's confirm.

```
hourly_check_1 <- combined_hourly[combined_hourly$Id==6962181067 & combined_hourly$Date=="4/12/2016",]
print(hourly_check_1)
```

```
## # A tibble: 10 x 8
##            Id ActivityHour         Date  Time  Calories StepTotal TotalIntensity
##         <dbl> <chr>                <chr> <chr>    <dbl>     <dbl>          <dbl>
##  1 6962181067 4/12/2016 12:00:00 ~ 4/12~ 12:0~       58        32              2
##  2 6962181067 4/12/2016 1:00:00 AM 4/12~ 1:00~       55         0              0
##  3 6962181067 4/12/2016 2:00:00 AM 4/12~ 2:00~       55         0              0
##  4 6962181067 4/12/2016 3:00:00 AM 4/12~ 3:00~       55         0              0
##  5 6962181067 4/12/2016 4:00:00 AM 4/12~ 4:00~       55         0              0
##  6 6962181067 4/12/2016 5:00:00 AM 4/12~ 5:00~       55         0              0
##  7 6962181067 4/12/2016 6:00:00 AM 4/12~ 6:00~       59        21              3
##  8 6962181067 4/12/2016 7:00:00 AM 4/12~ 7:00~       95       412             24
##  9 6962181067 4/12/2016 8:00:00 AM 4/12~ 8:00~       75       156             12
## 10 6962181067 4/12/2016 9:00:00 AM 4/12~ 9:00~      186      2812             76
## # i 1 more variable: AverageIntensity <dbl>
```

```
hourly_check_2 <- combined_hourly[combined_hourly$Id==8877689391 & combined_hourly$Date=="4/12/2016",]
print(hourly_check_2)
```

```
## # A tibble: 9 x 8
##            Id ActivityHour          Date  Time  Calories StepTotal TotalIntensity
##         <dbl> <chr>                 <chr> <chr>    <dbl>     <dbl>          <dbl>
```

```
## 1 8877689391 4/12/2016 12:00:00 AM 4/12~ 12:0~       73         0              0
## 2 8877689391 4/12/2016 1:00:00 AM  4/12~ 1:00~       73         0              0
## 3 8877689391 4/12/2016 2:00:00 AM  4/12~ 2:00~       73         0              0
## 4 8877689391 4/12/2016 3:00:00 AM  4/12~ 3:00~       73         0              0
## 5 8877689391 4/12/2016 4:00:00 AM  4/12~ 4:00~       73         0              0
## 6 8877689391 4/12/2016 5:00:00 AM  4/12~ 5:00~       73         0              0
## 7 8877689391 4/12/2016 6:00:00 AM  4/12~ 6:00~       96       209              7
## 8 8877689391 4/12/2016 7:00:00 AM  4/12~ 7:00~      169       964             26
## 9 8877689391 4/12/2016 8:00:00 AM  4/12~ 8:00~      136       834             17
## # i 1 more variable: AverageIntensity <dbl>
```

As expected, the data only goes to 9:00am on 4/12 for 6962181067 and 8am for 8877689391.

### 3c. Does any weight/fat loss or gain coincide with higher or lower levels of activity/intensity?

As in 3b, there isn't enough data in the weightLoginfo dataset to see a noteworthy change in weight/fat levels.

### 4a. Does it seem like people are wearing their smart devices all day or only when doing activities?

To answer this question, we'll check if the total number of minutes recorded in "daily_activity" is equal to the number of minutes in a day - 24 x 60 = 1440.

Let's add a column to daily_activity to record how many total minutes are included in each record and the difference between that number and 1440.

```
daily_activity$total_minutes <- daily_activity$VeryActiveMinutes + daily_activity$FairlyActiveMinutes +

daily_activity$untracked_minutes <- 1440 - daily_activity$total_minutes
length(daily_activity$untracked_minutes)
```

```
## [1] 457
```

```
print(daily_activity)
```

```
## # A tibble: 457 x 17
##            Id ActivityDate TotalSteps TotalDistance TrackerDistance
##         <dbl> <chr>              <dbl>         <dbl>           <dbl>
##  1 1503960366 3/25/2016         11004          7.11            7.11
##  2 1503960366 3/26/2016         17609         11.6            11.6
##  3 1503960366 3/27/2016         12736          8.53            8.53
##  4 1503960366 3/28/2016         13231          8.93            8.93
##  5 1503960366 3/29/2016         12041          7.85            7.85
##  6 1503960366 3/30/2016         10970          7.16            7.16
##  7 1503960366 3/31/2016         12256          7.86            7.86
##  8 1503960366 4/1/2016          12262          7.87            7.87
##  9 1503960366 4/2/2016          11248          7.25            7.25
## 10 1503960366 4/3/2016          10016          6.37            6.37
## # i 447 more rows
## # i 12 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>,
## #   total_minutes <dbl>, untracked_minutes <dbl>
```

```r
sum(daily_activity$untracked_minutes==0)
```

```
## [1] 233
```

```r
summary(daily_activity$untracked_minutes)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0       0       0     245     455    1399
```

Then let's break the records up into groups based on the amount of time that's not covered in daily_activity and visualize those groups in a graph.

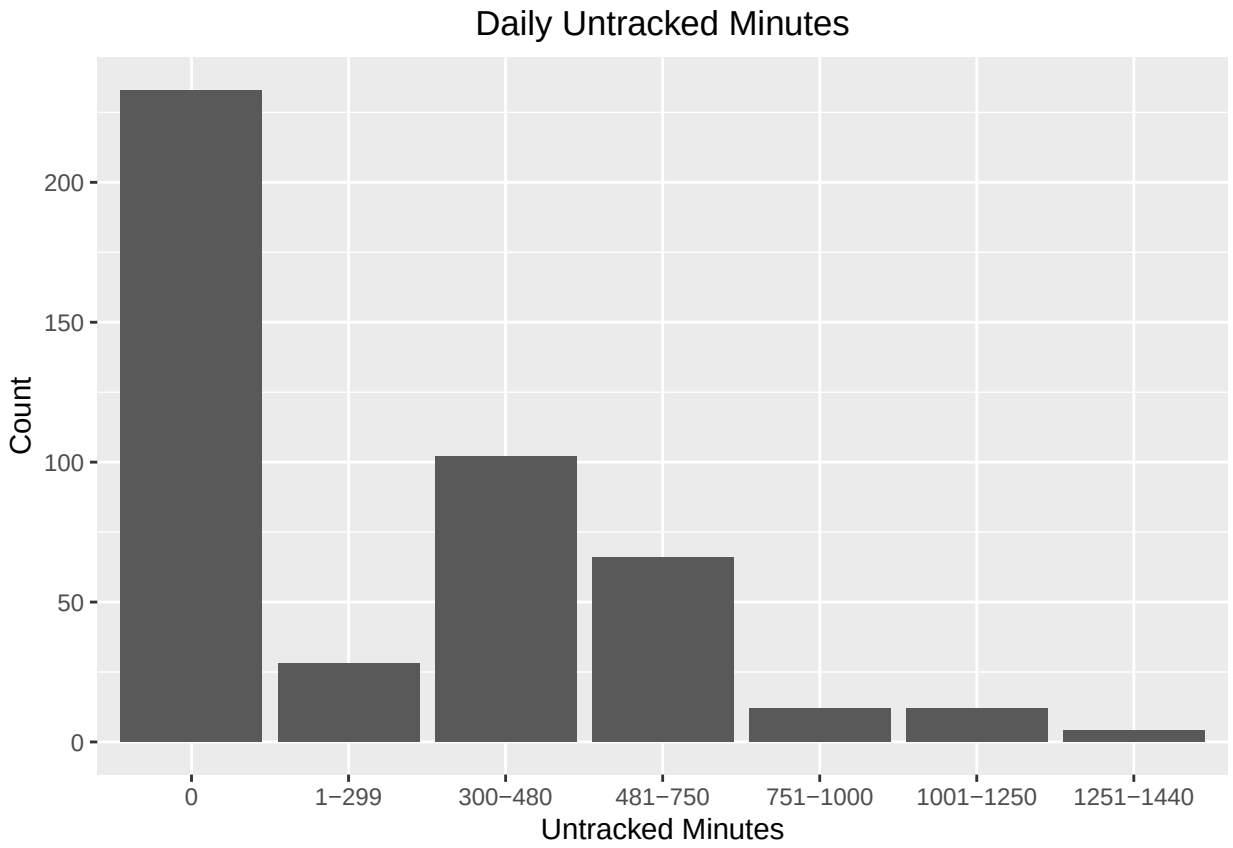
```r
untracked_buckets <- cut(daily_activity$untracked_minutes, breaks = c(0, 1, 299, 480, 750, 1000, 1250, 
print(table(untracked_buckets))
```

```
## untracked_buckets
##         0     1-299   300-480   481-750  751-1000 1001-1250 1251-1440
##       233        28       102        66        12        12         4
```

```r
bucket_df <- as.data.frame(table(untracked_buckets))
colnames(bucket_df) <- c("Minutes", "Count")
print(bucket_df)
```

```
##     Minutes Count
## 1         0   233
## 2     1-299    28
## 3   300-480   102
## 4   481-750    66
## 5  751-1000    12
## 6 1001-1250    12
## 7 1251-1440     4
```

```r
ggplot(bucket_df, aes(x = Minutes, y = Count)) +
  geom_col() +
  ggtitle("Daily Untracked Minutes") + xlab("Untracked Minutes") + ylab("Count") +
  theme(plot.title = element_text(hjust = 0.5))
```



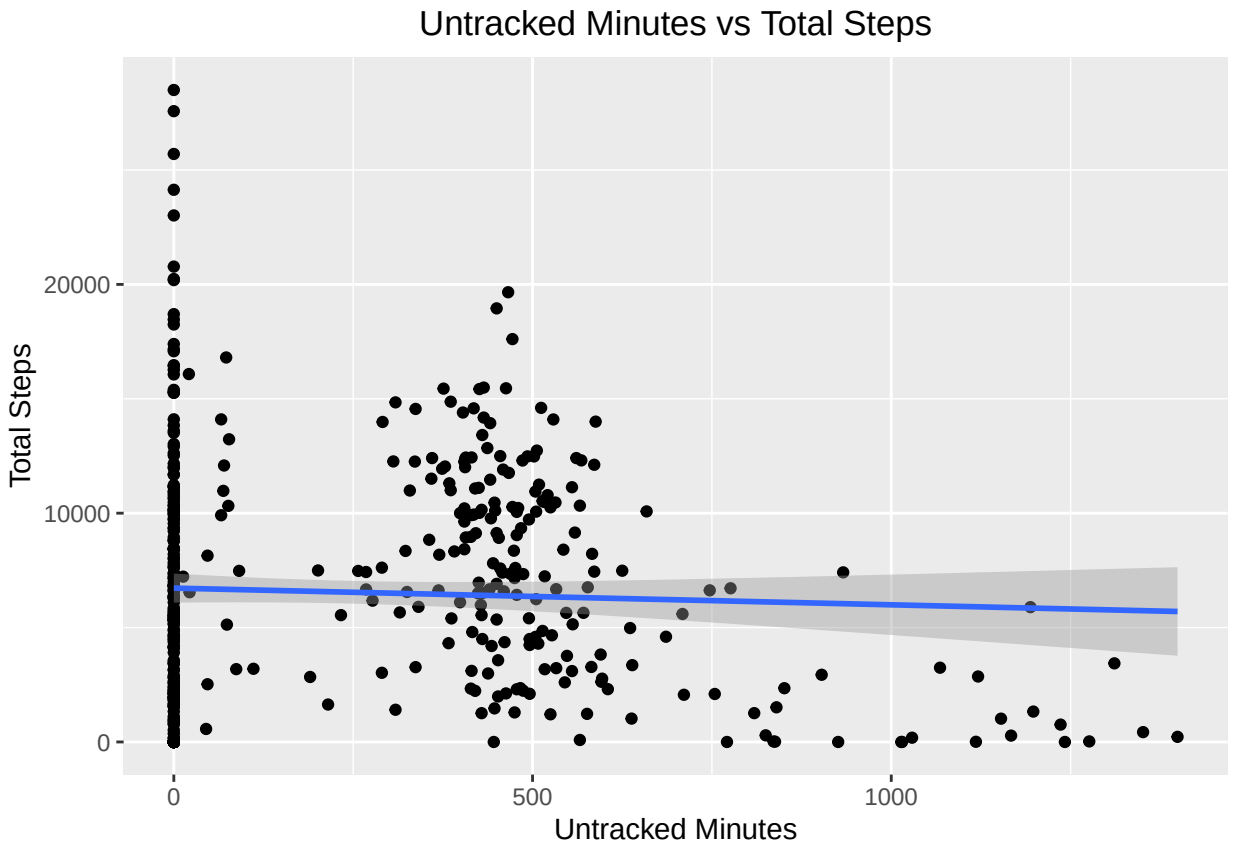

Of the 457 records in the daily_activity dataset, just over half - 233/50.98% - of those records have every minute of the day tracked and 57.11% have less than 5 hours untracked. However, almost a quarter - 102/22.32% - have 5-8 hours of untracked time, which is the range of time commonly spent asleep (in America) and another 20.57% (94) of users have more than 8 hours of untracked time. This leads me to a new question.

**4b. Are people who are not wearing their smart device all day less active than those who are?**

To look at this, we'll create scatterplots comparing untracked time and total steps/total calories.

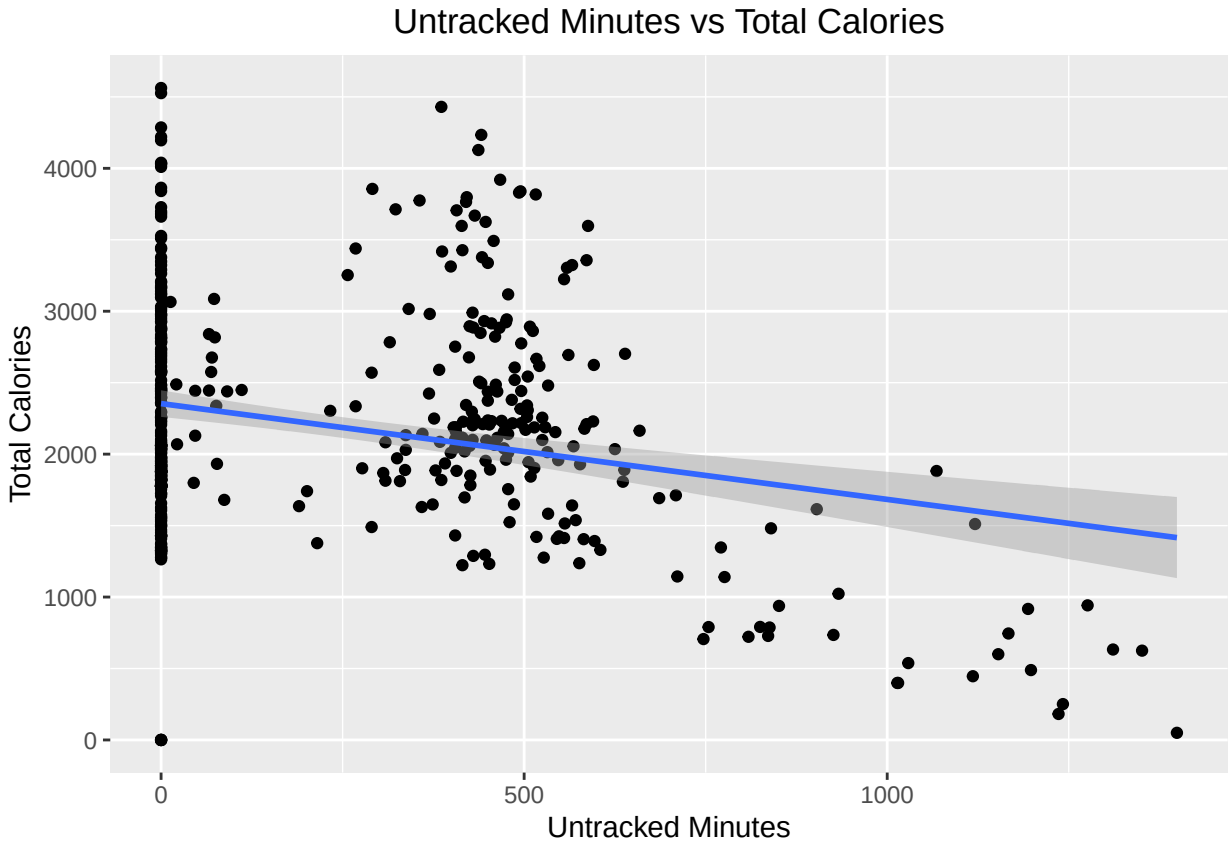
```
ggplot(daily_activity, aes(x = untracked_minutes, y = TotalSteps)) +
  geom_point() +
  geom_smooth(method = "lm") +
  ggtitle("Untracked Minutes vs Total Steps") + xlab("Untracked Minutes") + ylab("Total Steps") +
  theme(plot.title = element_text(hjust = 0.5))
```

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




```
ggplot(daily_activity, aes(x = untracked_minutes, y = Calories)) +
  geom_point() +
  geom_smooth(method = "lm") +
  ggtitle("Untracked Minutes vs Total Calories") + xlab("Untracked Minutes") + ylab("Total Calories") +
  theme(plot.title = element_text(hjust = 0.5))
```

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




As you would expect, the people with the highest amount of untracked minutes also have the lowest daily calories and steps. Given that you burn calories even when not active, the negative trendline in that graph is to be expected since those calories are not accounted for. So there is no indication that the group consisting of the people who mostly have around a night's sleep worth of untracked time are less active, based on the general spread of that datapoints in that group and the overall fairly flat trend lines in the graphs. This indicates to me that these people are mostly not wearing their smart device while sleeping.

**5. Do people tend to sleep better after days with higher amounts/intensities of activity?**

I can't find anywhere that explains the 1, 2, and 3 values in the minute_sleep dataset and while I can make assumptions based on what I see in the dataset I'd rather not do that.

**6a. What are the maxes (and mins? averages?) for each individual and do they line up/make sense?**

Let's create another dataframe, this time grouped by Id, to summarize the various metrics for each user.

```
by_user_activity <- combined_hourly %>%
  group_by(Id) %>%
  summarize(max_calories = max(Calories), max_steps = max(StepTotal), max_total_intensity = max(TotalIn
print(by_user_activity)
```

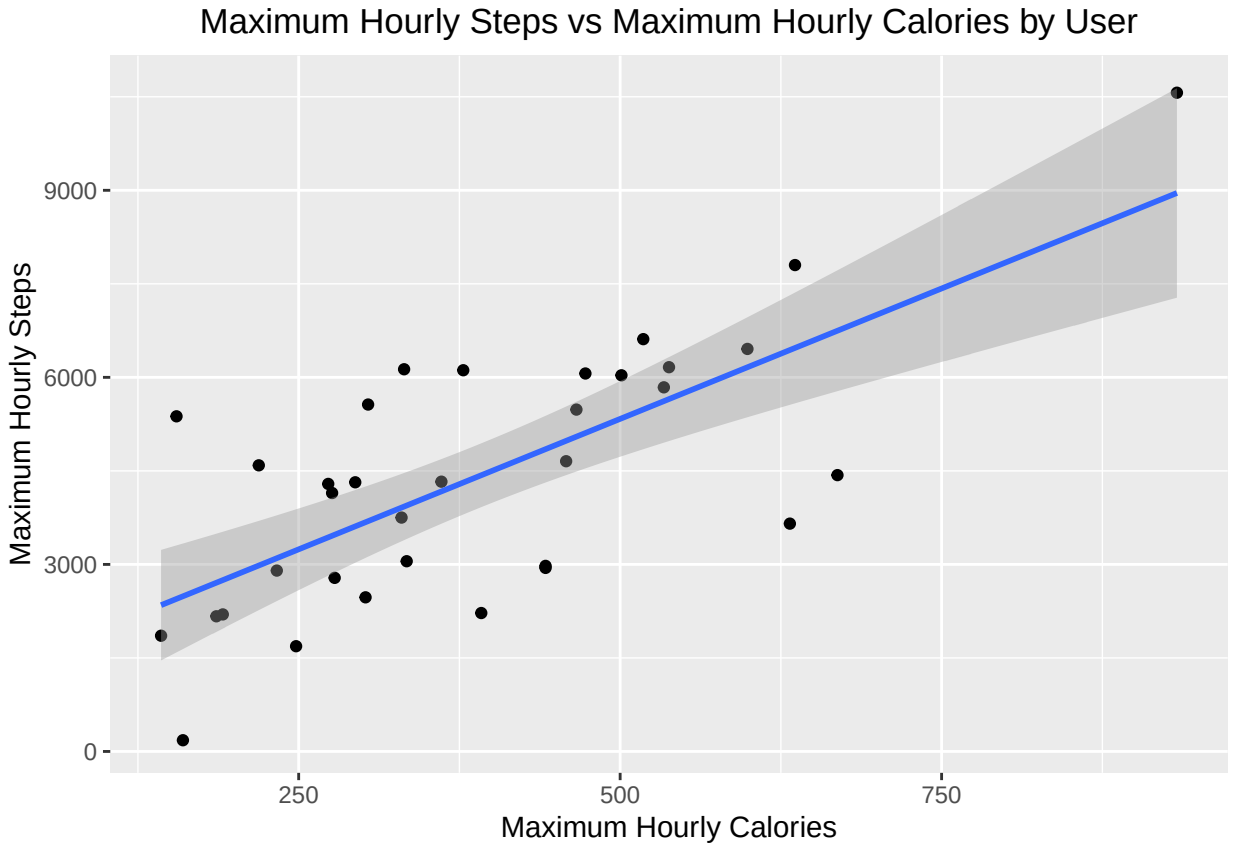
```
## # A tibble: 34 x 10
##            Id max_calories max_steps max_total_intensity avg_calories avg_steps
##         <dbl>        <dbl>     <dbl>               <dbl>        <dbl>     <dbl>
##  1 1503960366          304      5564                 149         81.1     548.
##  2 1624580081          155      5375                  66         58.7     191.
##  3 1644430081          458      4655                 129        127.      335.
##  4 1844505072          191      2198                  71         62.7      69.6
```

```
##  5 1927972279          534      5840                 174        109.      227.
##  6 2022484408          538      6163                 169        109.      531.
##  7 2026352035          143      1856                  53         61.7     188.
##  8 2320127002          186      2168                  60         61.2      72.3
##  9 2347167796          378      6114                 153         89.2     434.
## 10 2873212765          466      5483                 168         79.9     351.
## # i 24 more rows
## # i 4 more variables: avg_total_intensity <dbl>, min_calories <dbl>,
## #   min_steps <dbl>, min_total_intensity <dbl>
```

From a quick glance, the summary statistics seem to line up, but let's get a more concrete overall view using a few scatterplots. The minimum steps and total intensity is necessarily 0 for people wearing their devices while sleeping, so we'll ignore those for the visuals.

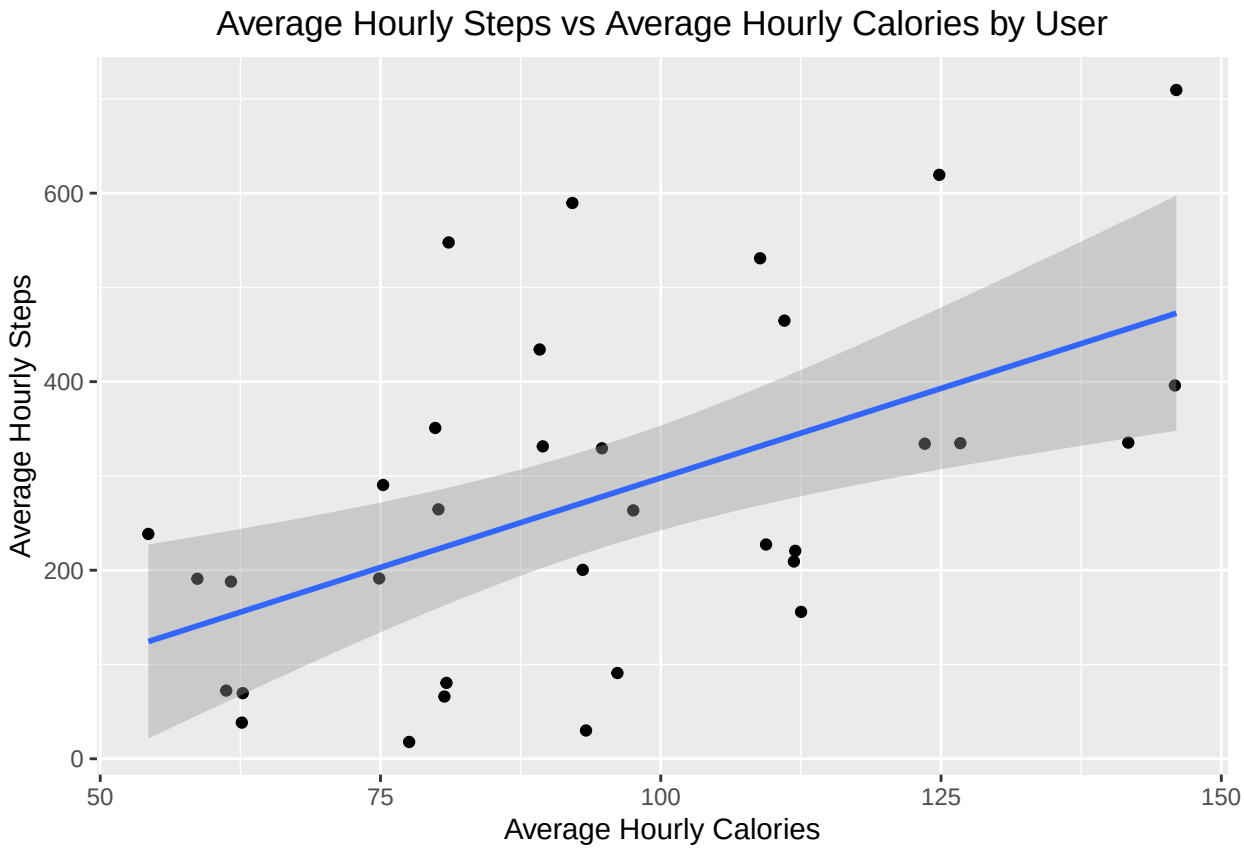
```
ggplot(by_user_activity, aes(x = max_calories, y = max_steps)) +
  geom_point() +
  geom_smooth(method = "lm") +
  ggtitle("Maximum Hourly Steps vs Maximum Hourly Calories by User") + xlab("Maximum Hourly Calories") +
  theme(plot.title = element_text(hjust = 0.5))
```

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



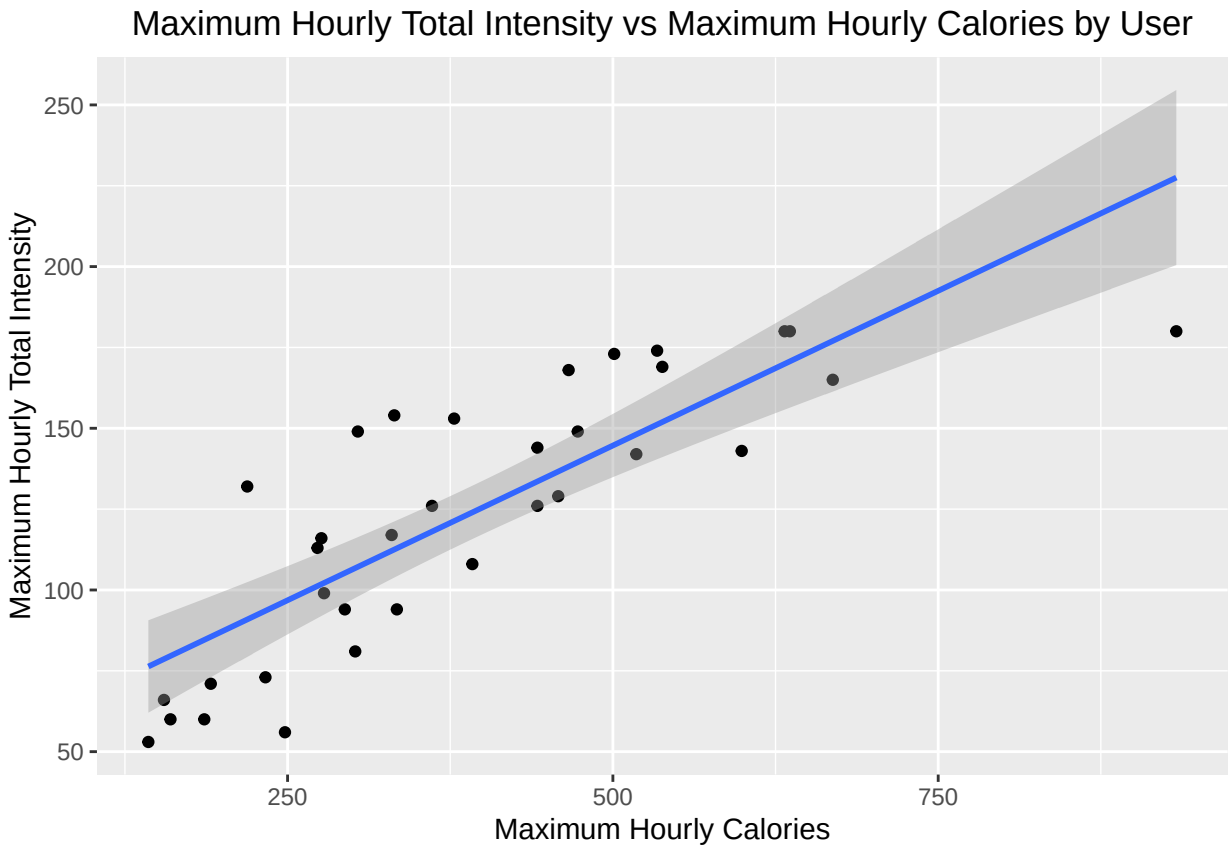

```
ggplot(by_user_activity, aes(x = avg_calories, y = avg_steps)) +
  geom_point() +
  geom_smooth(method = "lm") +
  ggtitle("Average Hourly Steps vs Average Hourly Calories by User") + xlab("Average Hourly Calories") +
  theme(plot.title = element_text(hjust = 0.5))
```

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



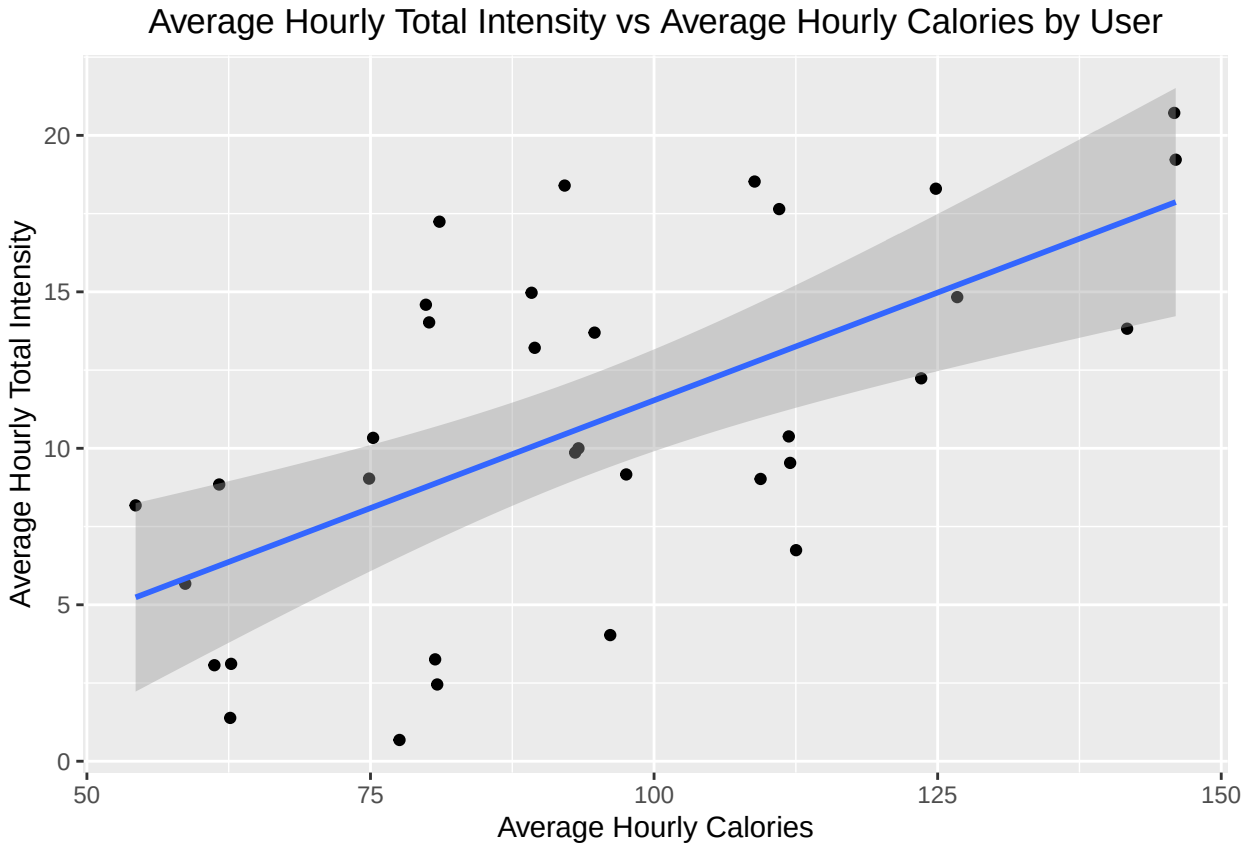

```
ggplot(by_user_activity, aes(x = max_calories, y = max_total_intensity)) +
  geom_point() +
  geom_smooth(method = "lm") +
  ggtitle("Maximum Hourly Total Intensity vs Maximum Hourly Calories by User") + xlab("Maximum Hourly Ca
  theme(plot.title = element_text(hjust = 0.5))
```

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



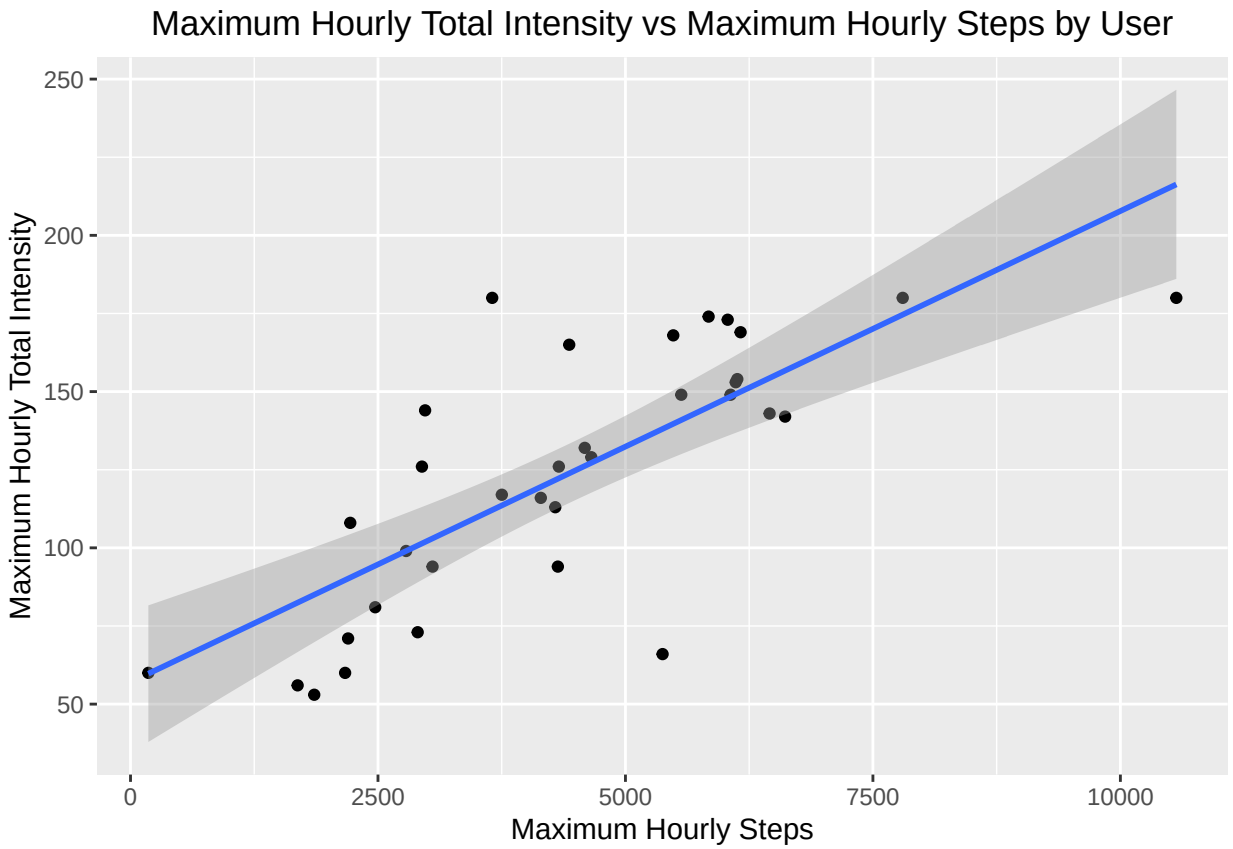

```
ggplot(by_user_activity, aes(x = avg_calories, y = avg_total_intensity)) +
  geom_point() +
  geom_smooth(method = "lm") +
  ggtitle("Average Hourly Total Intensity vs Average Hourly Calories by User") + xlab("Average Hourly Ca
  theme(plot.title = element_text(hjust = 0.5))
```

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



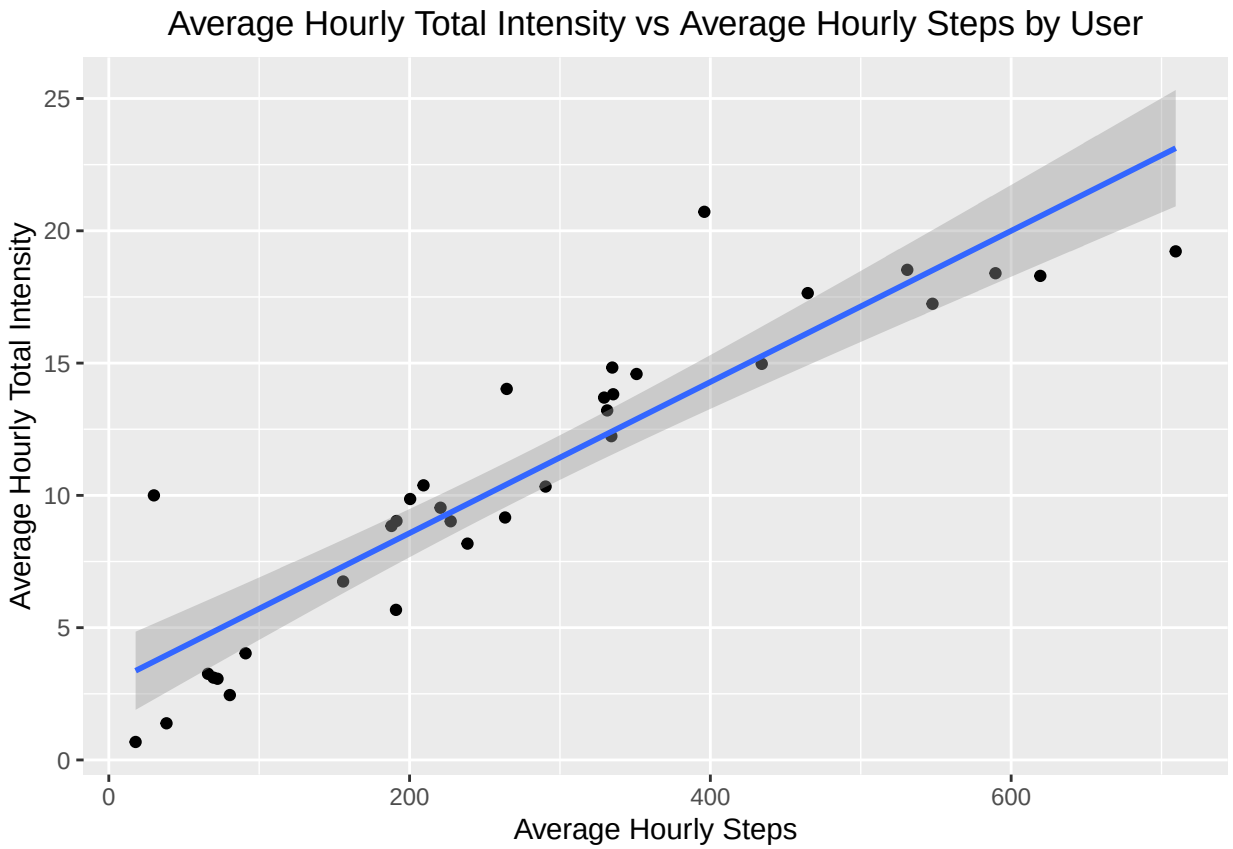

```
ggplot(by_user_activity, aes(x = max_steps, y = max_total_intensity)) +
  geom_point() +
  geom_smooth(method = "lm") +
  ggtitle("Maximum Hourly Total Intensity vs Maximum Hourly Steps by User") + xlab("Maximum Hourly Steps
  theme(plot.title = element_text(hjust = 0.5))
```

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




```
ggplot(by_user_activity, aes(x = avg_steps, y = avg_total_intensity)) +
  geom_point() +
  geom_smooth(method = "lm") +
  ggtitle("Average Hourly Total Intensity vs Average Hourly Steps by User") + xlab("Average Hourly Steps
  theme(plot.title = element_text(hjust = 0.5))
```

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




All of the graphs show the same general positive trends, as I would expect. The two "vs average hourly calories" graphs are more weakly grouped, but considering that higher levels of calories are burned by exercising in general this could indicate that many of the users are engaging in more physical activity than just walking/running. The points in the average hourly total intensity vs average hourly steps are surprisingly closely bunched around the trend line.

**6b. What does each user's general activity trendline look like? How many people increased in total time spent active or distance walked?**

Let's add a column to daily_activity to calculate every user's daily total active time.

```
daily_activity$total_activity_time <- daily_activity$VeryActiveMinutes + daily_activity$FairlyActiveMinu
```

**Categorize users based on activity level and look for trends** Let's add a column to by_user_activity to label users based on their average hourly number of steps. - Less than or equal to 100 - sedentary - Less than or equal to 200 - lightly active - Less than or equal to 400 - moderately active - Greater than 400 - very active

```
by_user_activity$activity_level <- case_when(
  by_user_activity$avg_steps <= 100 ~ "Sedentary",
  by_user_activity$avg_steps <= 200 ~ "Lightly Active",
  by_user_activity$avg_steps <= 400 ~ "Moderately Active",
  by_user_activity$avg_steps > 400 ~ "Very Active"
)

print(by_user_activity)
```

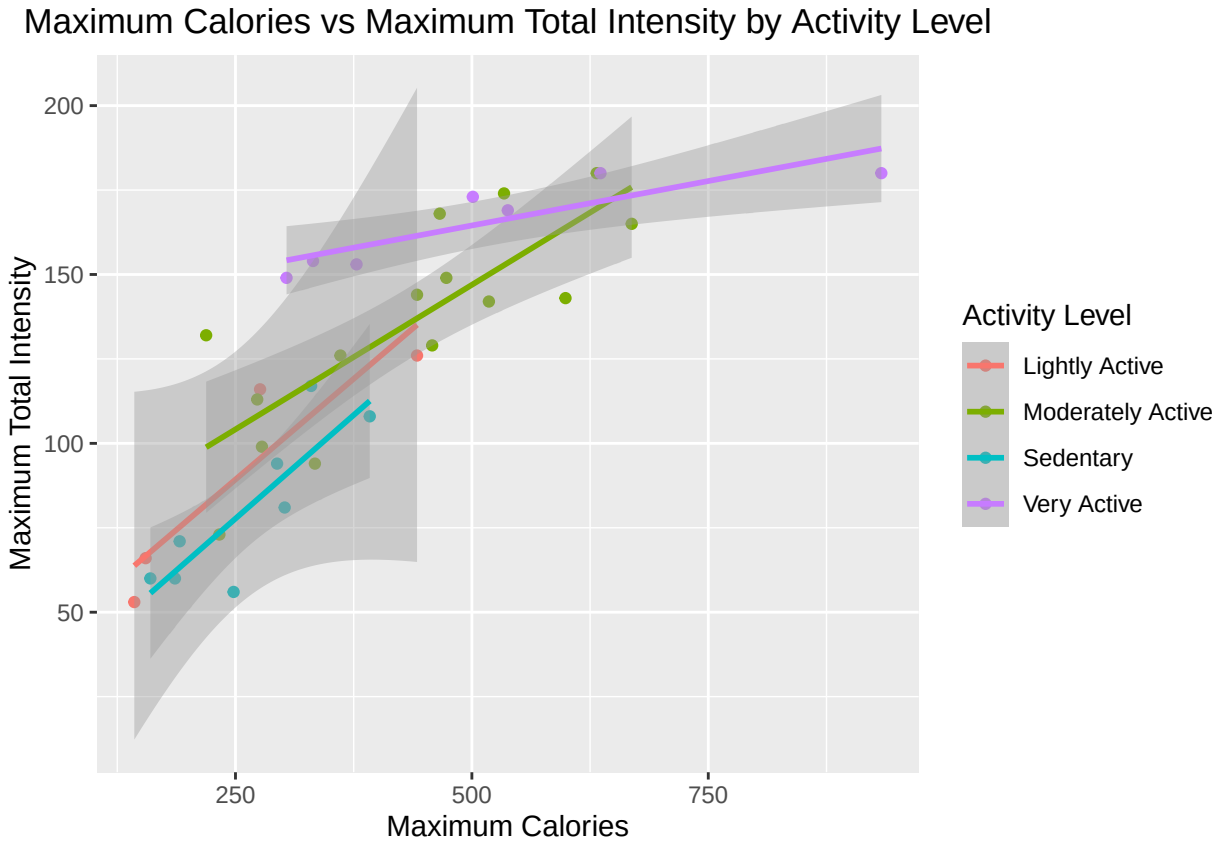
```
## # A tibble: 34 x 11
##            Id max_calories max_steps max_total_intensity avg_calories avg_steps
```

```
##         <dbl>        <dbl>     <dbl>                <dbl>        <dbl>     <dbl>
##  1 1503960366          304      5564                  149         81.1     548.
##  2 1624580081          155      5375                   66         58.7     191.
##  3 1644430081          458      4655                  129        127.      335.
##  4 1844505072          191      2198                   71         62.7      69.6
##  5 1927972279          534      5840                  174        109.      227.
##  6 2022484408          538      6163                  169        109.      531.
##  7 2026352035          143      1856                   53         61.7     188.
##  8 2320127002          186      2168                   60         61.2      72.3
##  9 2347167796          378      6114                  153         89.2     434.
## 10 2873212765          466      5483                  168         79.9     351.
## # i 24 more rows
## # i 5 more variables: avg_total_intensity <dbl>, min_calories <dbl>,
## #   min_steps <dbl>, min_total_intensity <dbl>, activity_level <chr>
```

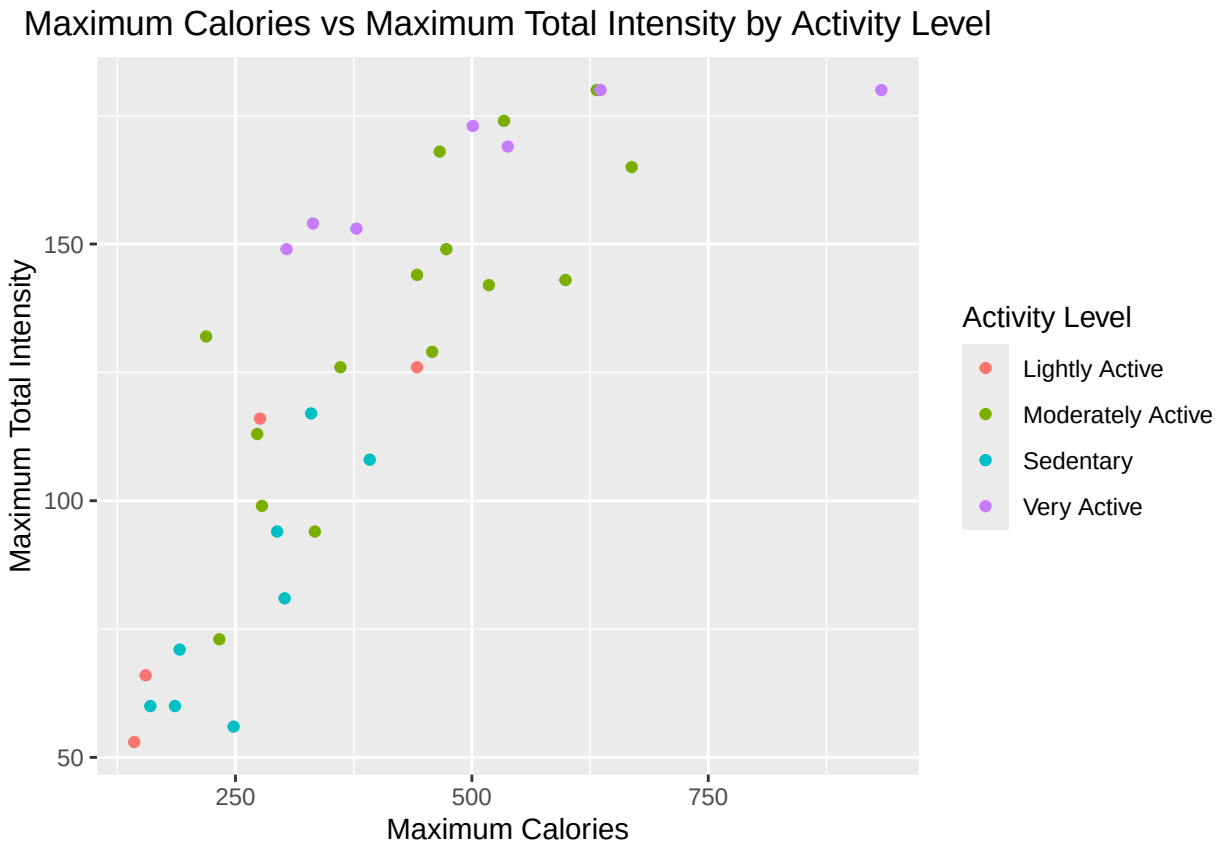
Now let's visualize the maxes and averages based on activity level, both with and without trendlines for clarity.

```
ggplot(by_user_activity, aes(x = max_calories, y = max_total_intensity, color = activity_level)) +
        geom_point() +
        geom_smooth(method = "lm") +
        labs(color="Activity Level") +
        ggtitle("Maximum Calories vs Maximum Total Intensity by Activity Level") + xlab("Maximum Calori
        theme(plot.title = element_text(hjust = 0.5))
```

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

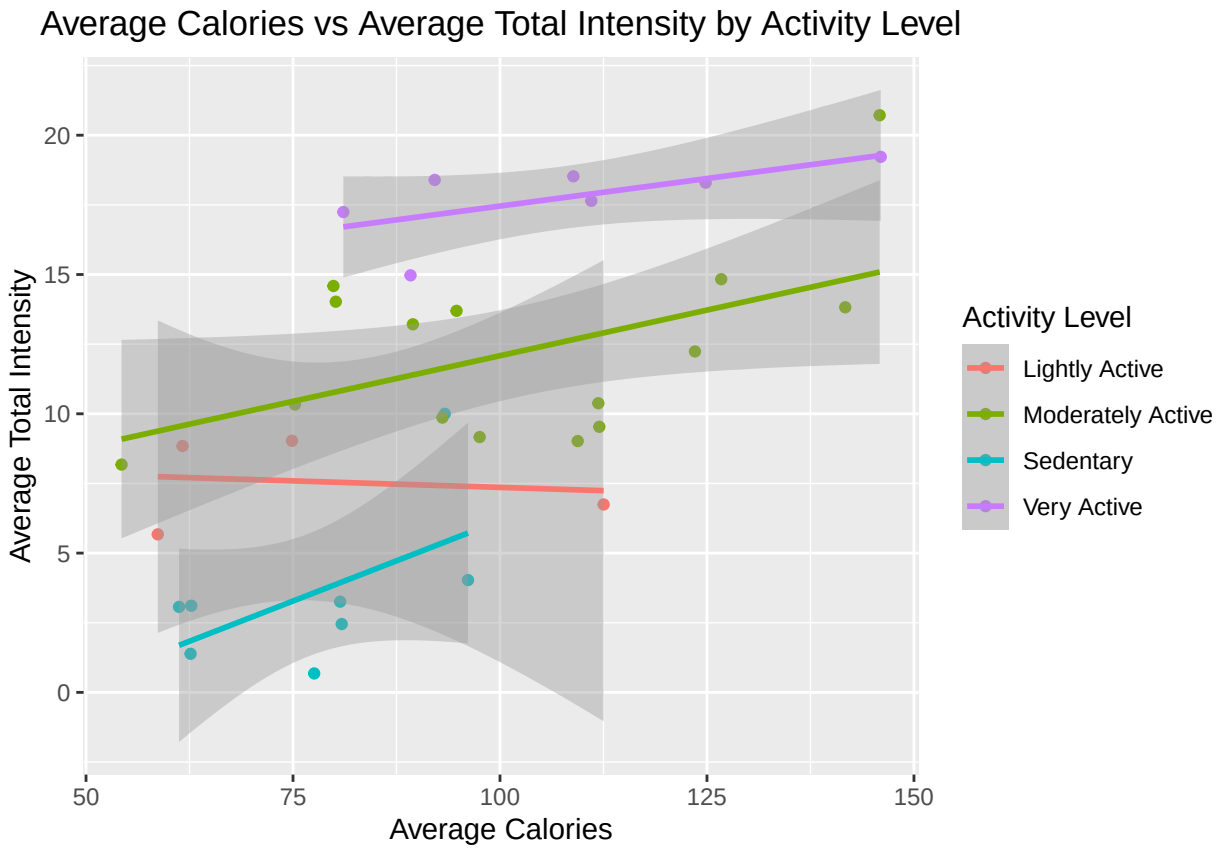


```
ggplot(by_user_activity, aes(x = max_calories, y = max_total_intensity, color = activity_level)) +
        geom_point() +
        labs(color="Activity Level") +
        ggtitle("Maximum Calories vs Maximum Total Intensity by Activity Level") + xlab("Maximum Calori
        theme(plot.title = element_text(hjust = 0.5))
```

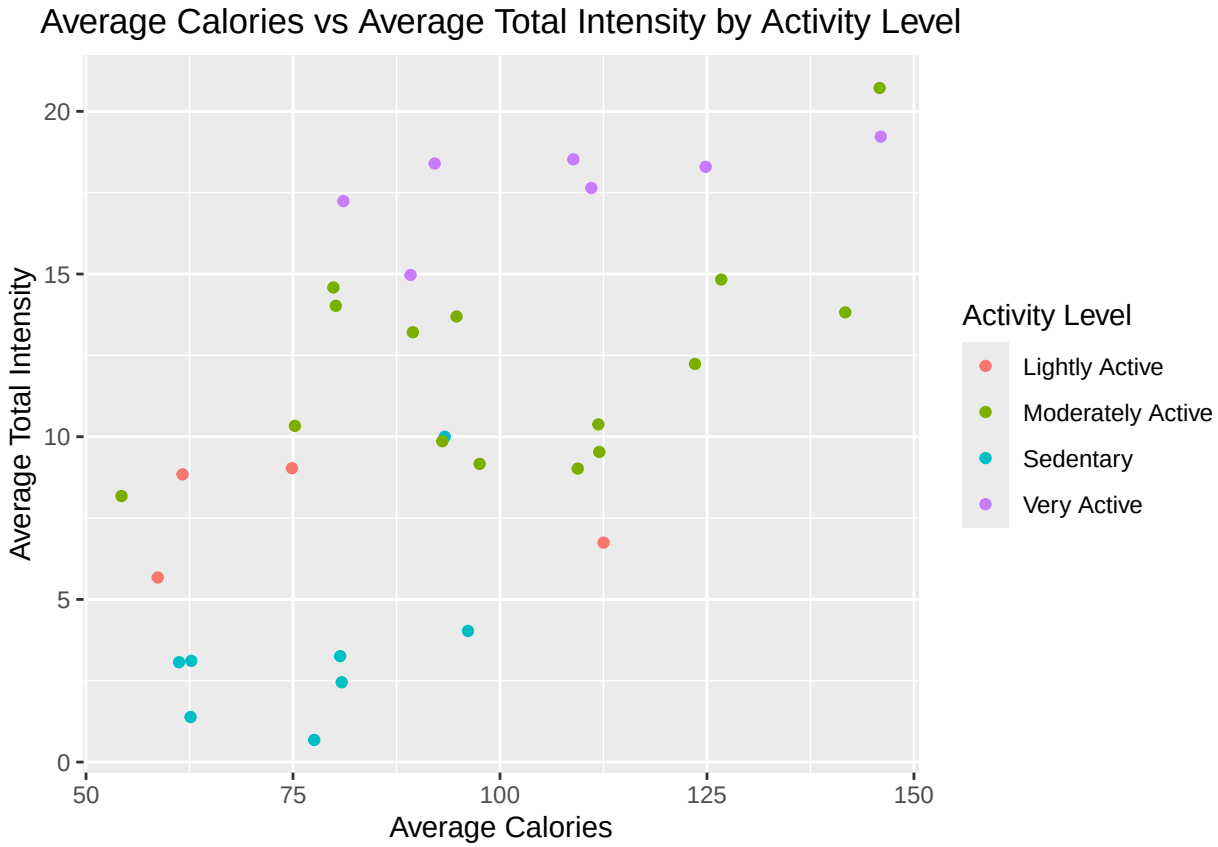



```
ggplot(by_user_activity, aes(x = avg_calories, y = avg_total_intensity, color = activity_level)) +
        geom_point() +
        geom_smooth(method = "lm") +
        labs(color="Activity Level") +
        ggtitle("Average Calories vs Average Total Intensity by Activity Level") + xlab("Average Calori
        theme(plot.title = element_text(hjust = 0.5))
```

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

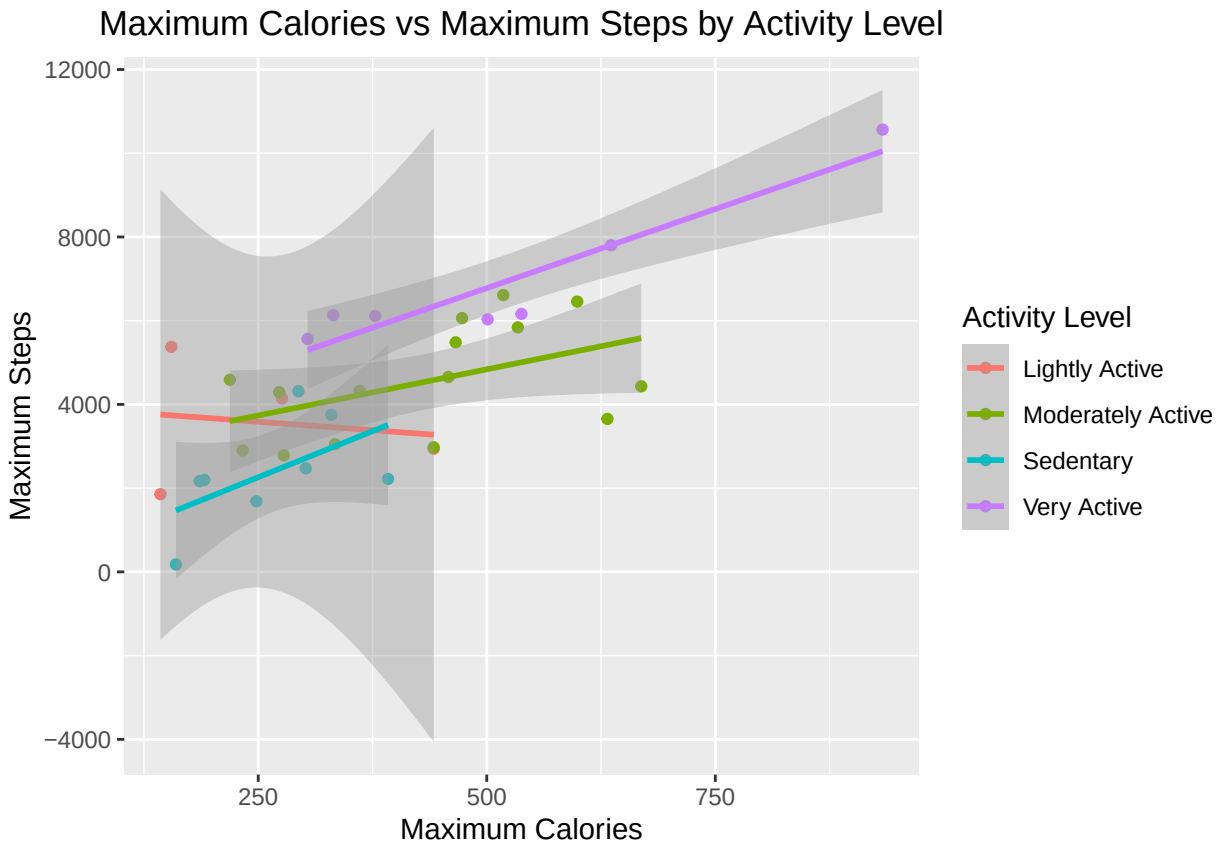



```
ggplot(by_user_activity, aes(x = avg_calories, y = avg_total_intensity, color = activity_level)) +
        geom_point() +
        labs(color="Activity Level") +
        ggtitle("Average Calories vs Average Total Intensity by Activity Level") + xlab("Average Calori
        theme(plot.title = element_text(hjust = 0.5))
```

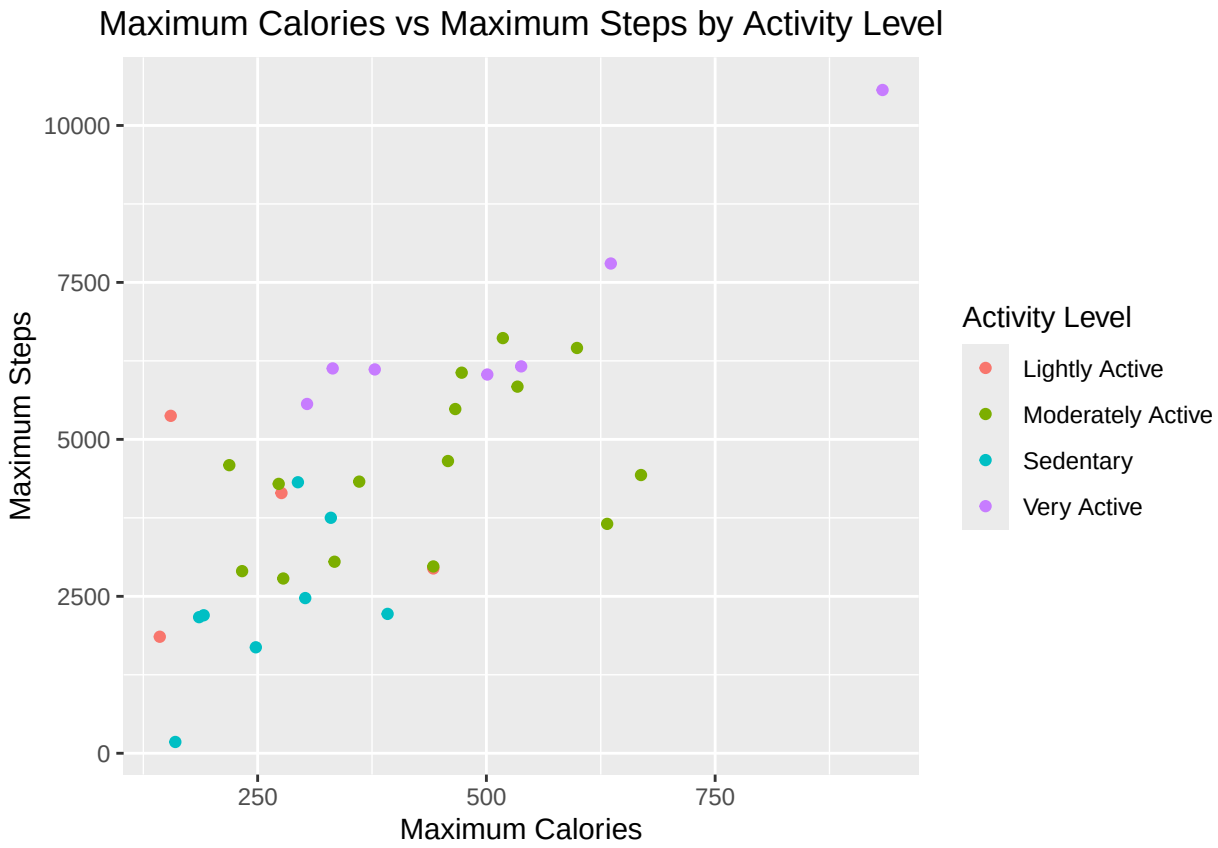



```
ggplot(by_user_activity, aes(x = max_calories, y = max_steps, color = activity_level)) +
         geom_point() +
         geom_smooth(method = "lm") +
         labs(color="Activity Level") +
         ggtitle("Maximum Calories vs Maximum Steps by Activity Level") + xlab("Maximum Calories") + yla
         theme(plot.title = element_text(hjust = 0.5))
```

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

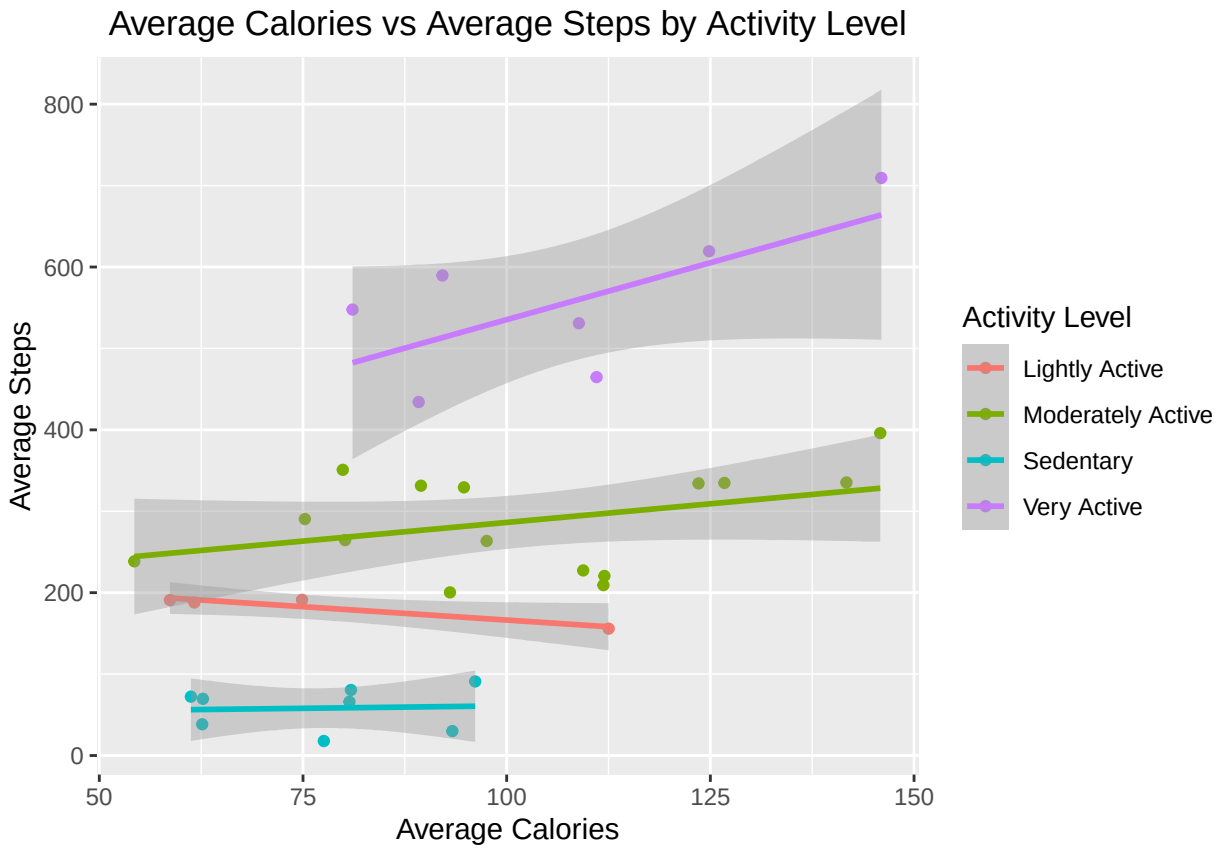



```
ggplot(by_user_activity, aes(x = max_calories, y = max_steps, color = activity_level)) +
         geom_point() +
         labs(color="Activity Level") +
         ggtitle("Maximum Calories vs Maximum Steps by Activity Level") + xlab("Maximum Calories") + yla
         theme(plot.title = element_text(hjust = 0.5))
```

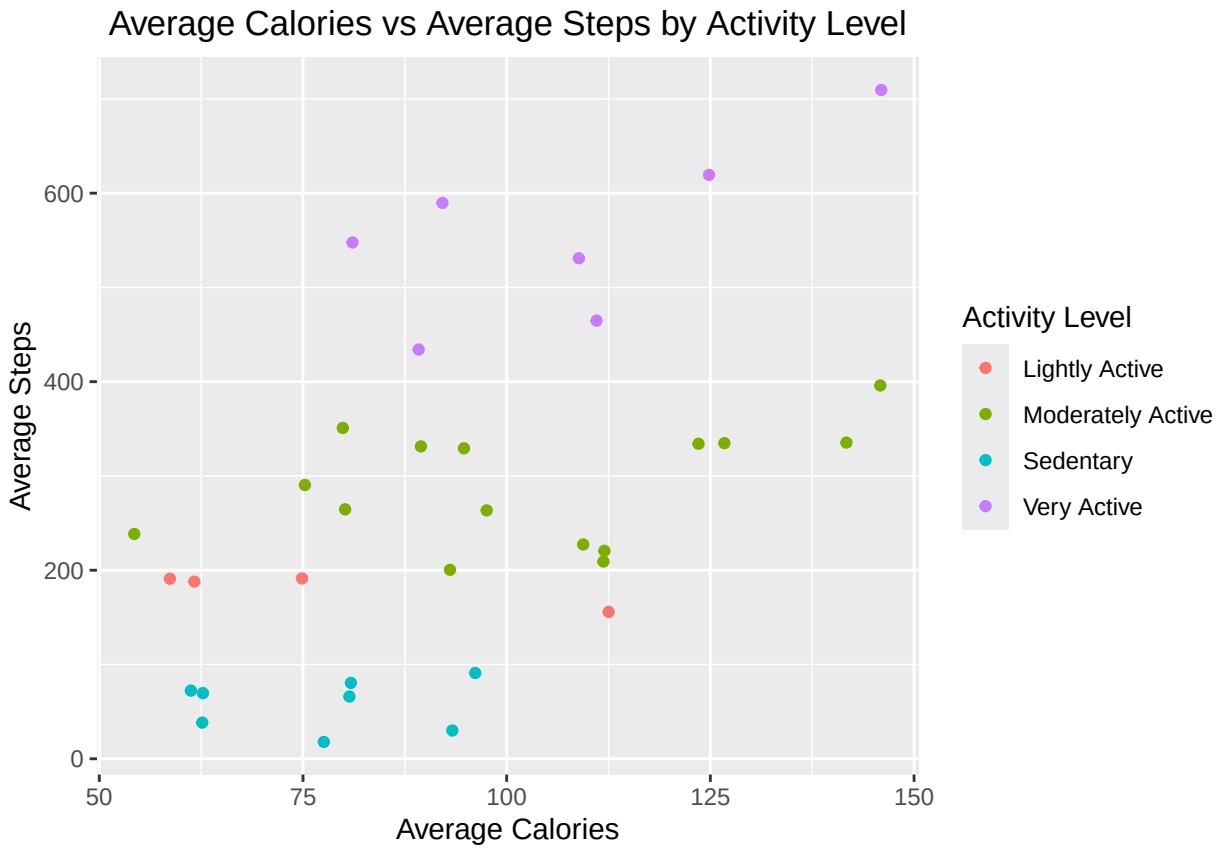



```
ggplot(by_user_activity, aes(x = avg_calories, y = avg_steps, color = activity_level)) +
         geom_point() +
         geom_smooth(method = "lm") +
         labs(color="Activity Level") +
         ggtitle("Average Calories vs Average Steps by Activity Level") + xlab("Average Calories") + yla
         theme(plot.title = element_text(hjust = 0.5))
```

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




```
ggplot(by_user_activity, aes(x = avg_calories, y = avg_steps, color = activity_level)) +
        geom_point() +
        labs(color="Activity Level") +
        ggtitle("Average Calories vs Average Steps by Activity Level") + xlab("Average Calories") + yla
        theme(plot.title = element_text(hjust = 0.5))
```




The graphs show that the users labelled with the same activity level are generally in the same area, as shown by the relatively low levels of overlap of the trendlines. Since the activity levels were assigned based on average steps and the graphs are based on intensity and calories, I believe this supports the conclusion that some of the users were involved in physical activities beyond walking. If we had that data, more robust conclusions could be drawn. One noteworthy mention is that the sedentary group has the steepest trendline in average total intensity over the time period represented in this data.

## SHARE

Guiding questions Were you able to answer the business questions? I believe so. Revisiting the main questions: 1. What are some trends in smart device usage? 2. How could these trends apply to Bellabeat customers? 3. How could these trends help influence Bellabeat marketing strategy?

We have clear trends in the provided data based on activity level, time of the day, etc. These trends could be similar to those of Bellabeat customers since they seem reasonable and like what I would expect from people in general. But they could allow Bellabeat to market both to specific groups - those looking to improve their fitness/increase the amount of time spent active (since there was a general increase in activity level) as well as people in general - "anyone can benefit from a tracker, from the sedentary to the marathon runners". The data also shows that sedentary users have the sharpest increase in average

What story does your data tell? The data shows that people who start tracking their activity levels tend to become more active over time and that you don't have to already be an active person to start tracking/exercising.

How do your findings relate to your original question? My findings directly address the questions that I originally came up with as I started looking at the data.

Who is your audience? What is the best way to communicate with them? My audience is the executive team of Bellabeat. The best way to communicate with them to is put things plainly and be able to back it up with visualizations and a good story. One of the cofounders is a mathematician and presumably has a

deeper understanding of some of the methodology used to analyze the data or at least is better equipped to understand the meanings of the visualizations.

Can data visualization help you share your findings? Yes, the visualizations clearly show the trends and findings from the analysis and will make it easier for stakeholders to understand the findings at a glance.

Is your presentation accessible to your audience? I believe so. The findings are not particularly technical and the visualizations are detailed but not bloated with information.

## ACT

High-level recommendations and takeaways: I understand the data is from FitBit users, not Bellabeat users, but enabling users to log/look at the data for more than just steps taken will allow for more robust data and deeper and more thorough analysis. This will then translate to being able to make recommendations for/market to more specific groups.

It might be a good idea to focus marketing towards people who want to exercise more or become more active, citing the trend of users tending to increase in activity levels in even as short a period of time as a month.

Can also add to marketing the encouraging thought that you don't have to be someone who is currently active to become someone who is more active - users span all levels of activity, from sedentary to marathon running. This is probably especially good given the increase in work-from-home positions.