BellaBeatAnalysis_RMarkdown

2024-04-03

How Can a Wellness Technology Company Play It Smart?

Urška Sršen and Sando Mur founded Bellabeat, a high-tech company that manufactures health-focused smart products in 2013. They develop beautifully designed technology that informs and inspires women around the world. Collecting data on activity, sleep, stress, and reproductive health has allowed Bellabeat to empower women with knowledge about their own health and habits.

Bellabeat's products:

- Bellabeat app: The Bellabeat app provides users with health data related to their activity, sleep, stress, menstrual cycle, and mindfulness habits. This data can help users better understand their current habits and make healthy decisions. The Bellabeat app connects to their line of smart wellness products.
- Leaf: Bellabeat's classic wellness tracker can be worn as a bracelet, necklace, or clip. The Leaf tracker connects to the Bellabeat app to track activity, sleep, and stress.
- Time: This wellness watch combines the timeless look of a classic timepiece with smart technology to track user activity, sleep, and stress. The Time watch connects to the Bellabeat app to provide you with insights into your daily wellness.
- Spring: This is a water bottle that tracks daily water intake using smart technology to ensure that you are appropriately hydrated throughout the day. The Spring bottle connects to the Bellabeat app to track your hydration levels.
- Bellabeat membership: Bellabeat also offers a subscription-based membership program for users. Membership gives users 24/7 access to fully personalized guidance on nutrition, activity, sleep, health and beauty, and mindfulness based on their lifestyle and goals.

Business Task

In this analysis of smart device usage data the business task is to gain insight into how people are already using their smart devices in order to make recommendations for how these trends can be used by the Bellabeat marketing team.

I focused on the Bellabeat membership to find areas that could be improved allowing it to grow the features it offers their users, and also attract more users to it.

Database

For this analysis I used the "FitBit Fitness Tracker Data" data set, that is publicly available in Kaggle. It contains personal fitness tracker data from thirty fitbit users who consented to the submission of personal tracker data, including minute-level output for physical activity, heart rate, and sleep monitoring. It includes information about daily activity, steps, and heart rate that can be used to explore users' habits. It consists of 2 folders with 29 csv files, which some are duplicates, with data organized by minute, hour and day on different files. This data has limitations mainly because of the year it was collected and the sample data size. This dataset was generated by

respondents to a distributed survey via Amazon Mechanical Turk between 03.12.2016-05.12.2016. Citation: Furberg, R., Brinton, J., Keating, M., & Ortiz, A. (2016). Crowd-sourced Fitbit datasets 03.12.2016-05.12.2016 [Data set]. Zenodo. https://doi.org/10.5281/zenodo.53894 (https://doi.org/10.5281/zenodo.53894)

Process

During this process I used Google Sheets to check, clean and format the data. Then I used R for the analysis and visualizations. I decided to use R because it gives me the option to easily replicate the analysis, which we'd be useful if the data set is updated for example. It also allows me to keep a detailed summary of the steps I took using an R Markdown file.

First I downloaded the files from Kaggle, unzipped them and uploaded them to Google Sheets to process it and get it ready for analysis. Of the data set files I used:
dailyActivity_merged.csv
dailyCalories_merged.csv
dailyIntensities_merged.csv
dailySteps_merged.csv
sleepDay_merged.csv
weightLogInfo_m
hourlyCalories_merged.csv
hourlyIntensities_merged.csv
hourlySteps_merged.csv

In Google Sheets I divided the Date-Time column in date and time separately and formatted them accordingly. For the hourly files the time column I formatted it as integers to make the visualizations more clear using a format of 1-24 hours. Then I looked for duplicated data and data that might have been entered incorrectly so I could correct it to the right format. After checking all the data was good for analysis I downloaded the files as csv format.

Now I was ready to use R for the analysis of the data. I logged into R Studio, an online tool I used for this assignment, and started with an R file and a R Markdown file to keep track of the steps I took.

Analysis

Set up the enviroment

In order to do the analysis first I needed to install and load packages into the environment.

```
library(ggplot2)
library(readr)
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

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

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

Uploading dataset

I'm using the files that I cleaned and prepared with Google Sheets. I uploaded them in the RStudio cloud, which is the tool I'm using for this analysis.

```
dailyActivity <- read_csv("C:/Users/sofia/Desktop/Portfolio Projects/BellaBeat/CleanDatasets/dai
lyActivity.csv")
```

```
## Rows: 940 Columns: 15
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (1): Date
## 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.
```

```
dailyCalories <- read_csv("C:/Users/sofia/Desktop/Portfolio Projects/BellaBeat/CleanDatasets/dai
lyCalories.csv")
```

```
## Rows: 940 Columns: 3
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (1): Date
## 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.
```

```
dailyIntensities <- read_csv("C:/Users/sofia/Desktop/Portfolio Projects/BellaBeat/CleanDatasets/
dailyIntensities.csv")
```

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

```
dailySleep <- read_csv("C:/Users/sofia/Desktop/Portfolio Projects/BellaBeat/CleanDatasets/dailyS
leep.csv")
```

```
## Rows: 413 Columns: 5
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr (1): Date
## dbl (4): Id, TotalSleepRecords, TotalMinutesAsleep, TotalTimeInBed
##
## 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.
```

```
dailySteps <- read_csv("C:/Users/sofia/Desktop/Portfolio Projects/BellaBeat/CleanDatasets/dailySteps.csv")
```

```
## Rows: 940 Columns: 3
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr (1): Date
## 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.
```

```
hourlyCalories <- read_csv("C:/Users/sofia/Desktop/Portfolio Projects/BellaBeat/CleanDatasets/hourlyCalories.csv")
```

```
## Rows: 22099 Columns: 4
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr (2): Date, Time
## 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.
```

```
hourlyIntensities <- read_csv("C:/Users/sofia/Desktop/Portfolio Projects/BellaBeat/CleanDatasets/hourlyIntensities.csv")
```

```
## Rows: 22099 Columns: 5
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr (2): Date, Time
## 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.
```

```
hourlySteps <- read_csv("C:/Users/sofia/Desktop/Portfolio Projects/BellaBeat/CleanDatasets//hourlySteps.csv")
```

```
## Rows: 22099 Columns: 4
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr (2): Date, Time
## 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.
```

```
weightLogInfo <- read_csv("C:/Users/sofia/Desktop/Portfolio Projects/BellaBeat/CleanDatasets/wei
ghtLogInfo.csv")
```

```
## Rows: 67 Columns: 9
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr  (1): Date
## dbl  (6): Id, WeightKg, WeightPounds, Fat, BMI, LogId
## lgl  (1): IsManualReport
## time (1): Time
##
## 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.
```

Check datasets

I needed to check that the datasets were loaded correctly by using the head function. This function is also useful to get familiar with the data sets. In this case I already knew the column's name and type because I got familiar with the datasets in spreadsheets while I was cleaning and organizing them before starting this analysis using R Studio.

```
head(dailyActivity)
```

```
## # A tibble: 6 × 15
##         Id Date  TotalSteps TotalDistance TrackerDistance LoggedActivitiesDist…¹
##      <dbl> <chr>      <dbl>         <dbl>           <dbl>                   <dbl>
## 1   1.50e9 4/12…      13162          8.5             8.5                        0
## 2   1.50e9 4/13…      10735          6.97            6.97                       0
## 3   1.50e9 4/14…      10460          6.74            6.74                       0
## 4   1.50e9 4/15…       9762          6.28            6.28                       0
## 5   1.50e9 4/16…      12669          8.16            8.16                       0
## 6   1.50e9 4/17…       9705          6.48            6.48                       0
## # ℹ abbreviated name: ¹LoggedActivitiesDistance
## # ℹ 9 more variables: VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>
```

```
head(dailyCalories)
```

```
## # A tibble: 6 × 3
##           Id Date      Calories
##        <dbl> <chr>        <dbl>
## 1 1503960366 4/12/2016     1985
## 2 1503960366 4/13/2016     1797
## 3 1503960366 4/14/2016     1776
## 4 1503960366 4/15/2016     1745
## 5 1503960366 4/16/2016     1863
## 6 1503960366 4/17/2016     1728
```

```
head(dailyIntensities)
```

```
## # A tibble: 6 × 10
##           Id Date      SedentaryMinutes LightlyActiveMinutes FairlyActiveMinutes
##        <dbl> <chr>                <dbl>                <dbl>                <dbl>
## 1 1503960366 4/12/2016              728                  328                   13
## 2 1503960366 4/13/2016              776                  217                   19
## 3 1503960366 4/14/2016             1218                  181                   11
## 4 1503960366 4/15/2016              726                  209                   34
## 5 1503960366 4/16/2016              773                  221                   10
## 6 1503960366 4/17/2016              539                  164                   20
## # ℹ 5 more variables: VeryActiveMinutes <dbl>, SedentaryActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, moderatelyActiveDistance <dbl>,
## #   VeryActiveDistance <dbl>
```

```
head(dailySleep)
```

```
## # A tibble: 6 × 5
##           Id Date      TotalSleepRecords TotalMinutesAsleep TotalTimeInBed
##        <dbl> <chr>                 <dbl>              <dbl>          <dbl>
## 1 1503960366 4/12/2016                 1                327            346
## 2 1503960366 4/13/2016                 2                384            407
## 3 1503960366 4/15/2016                 1                412            442
## 4 1503960366 4/16/2016                 2                340            367
## 5 1503960366 4/17/2016                 1                700            712
## 6 1503960366 4/19/2016                 1                304            320
```

```
head(dailySteps)
```

```
## # A tibble: 6 × 3
##           Id Date      StepTotal
##        <dbl> <chr>         <dbl>
## 1 1503960366 4/12/2016     13162
## 2 1503960366 4/13/2016     10735
## 3 1503960366 4/14/2016     10460
## 4 1503960366 4/15/2016      9762
## 5 1503960366 4/16/2016     12669
## 6 1503960366 4/17/2016      9705
```

```
head(hourlyCalories)
```

```
## # A tibble: 6 × 4
##           Id Date      Time  Calories
##        <dbl> <chr>     <chr>    <dbl>
## 1 1503960366 4/12/2016 00          81
## 2 1503960366 4/12/2016 01          61
## 3 1503960366 4/12/2016 02          59
## 4 1503960366 4/12/2016 03          47
## 5 1503960366 4/12/2016 04          48
## 6 1503960366 4/12/2016 05          48
```

```
head(hourlyIntensities)
```

```
## # A tibble: 6 × 5
##           Id Date      Time  TotalIntensity AverageIntensity
##        <dbl> <chr>     <chr>          <dbl>            <dbl>
## 1 1503960366 4/12/2016 00                20            0.333
## 2 1503960366 4/12/2016 01                 8            0.133
## 3 1503960366 4/12/2016 02                 7            0.117
## 4 1503960366 4/12/2016 03                 0            0
## 5 1503960366 4/12/2016 04                 0            0
## 6 1503960366 4/12/2016 05                 0            0
```

```
head(hourlySteps)
```

```
## # A tibble: 6 × 4
##           Id Date      Time  StepTotal
##        <dbl> <chr>     <chr>     <dbl>
## 1 1503960366 4/12/2016 00          373
## 2 1503960366 4/12/2016 01          160
## 3 1503960366 4/12/2016 02          151
## 4 1503960366 4/12/2016 03            0
## 5 1503960366 4/12/2016 04            0
## 6 1503960366 4/12/2016 05            0
```

```
head(weightLogInfo)
```

```
## # A tibble: 6 × 9
##       Id Date  Time     WeightKg WeightPounds   Fat   BMI IsManualReport   LogId
##    <dbl> <chr> <time>      <dbl>        <dbl> <dbl> <dbl> <lgl>            <dbl>
## 1 1.50e9 5/2/… 23:59:59     52.6         116.    22  22.6 TRUE           1.46e12
## 2 1.50e9 5/3/… 23:59:59     52.6         116.    NA  22.6 TRUE           1.46e12
## 3 1.93e9 4/13… 01:08:52    134.          294.    NA  47.5 FALSE          1.46e12
## 4 2.87e9 4/21… 23:59:59     56.7         125.    NA  21.5 TRUE           1.46e12
## 5 2.87e9 5/12… 23:59:59     57.3         126.    NA  21.7 TRUE           1.46e12
## 6 4.32e9 4/17… 23:59:59     72.4         160.    25  27.5 TRUE           1.46e12
```

Create data frame

I needed to create a data frame that would allow me to compare the number of people that participated in each category of the data collected for the data sets. After creating it I can use it to visualize the results.

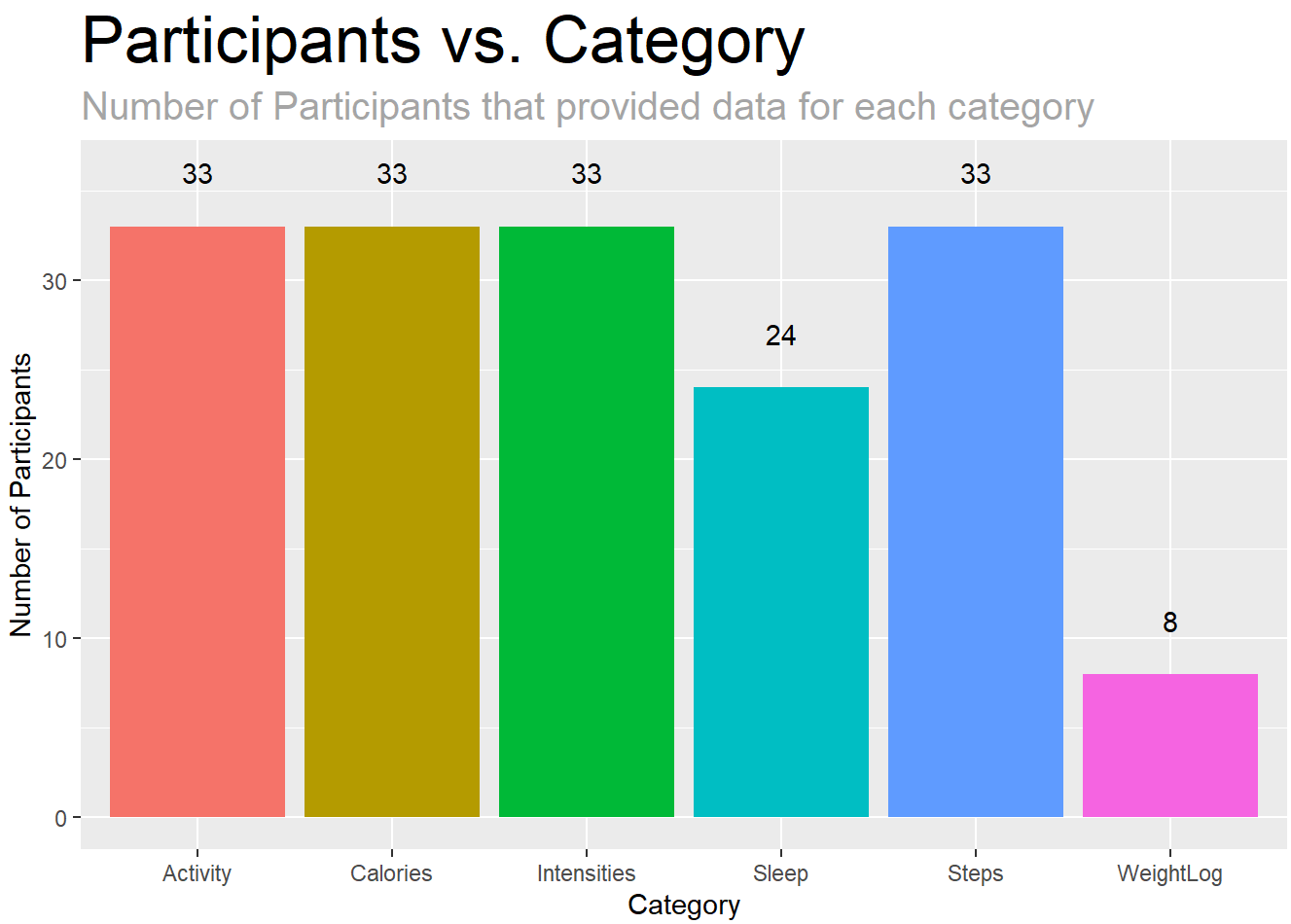
```
participants <- data.frame(Category = c("Activity", "Calories", "Intensities", "Sleep", "Steps",
"WeightLog"),
                           Number = c(n_distinct(dailyActivity$Id),
                                      n_distinct(dailyCalories$Id),
                                      n_distinct(dailyIntensities$Id),
                                      n_distinct(dailySleep$Id),
                                      n_distinct(dailySteps$Id),
                                      n_distinct(weightLogInfo$Id)
                                      )
                          )
head(participants)
```

```
##      Category Number
## 1    Activity     33
## 2    Calories     33
## 3 Intensities     33
## 4       Sleep     24
## 5       Steps     33
## 6   WeightLog      8
```

Participants in each Category

I created a chart in order to see how many people participated in each category of data collected for the database I'm using.

```
ggplot(data=participants, aes(x=Category, y=Number, fill=Category))+
  geom_col()+
  labs(title = "Participants vs. Category",
       subtitle = "Number of Participants that provided data for each category",
       y = "Number of Participants")+
  geom_text(aes(label = signif(Number)), nudge_y = 3)+
  theme(legend.position="none", panel.spacing.x = unit(0, "npc"),
        plot.title = element_text(size = 25),
        plot.subtitle = element_text(size = 15, colour = "darkgrey"))
```



As we can see in the chart there were less than 10 people in the weightLog collection of data, this means that the dataset is too small to make predictions about the uses. Also it tells us that this feature in the app is not being used that much, which means it's something that maybe users need more information about, there might be users not using it because they don't know how to use it or are not aware that this feature even exists.

Merge datasets

In order to visualize relationships between data in different datasets I needed to merge them first.

```
merged_daily <- merge(dailySleep, dailyActivity, by=c('Id', 'Date'))

head(merged_daily)
```

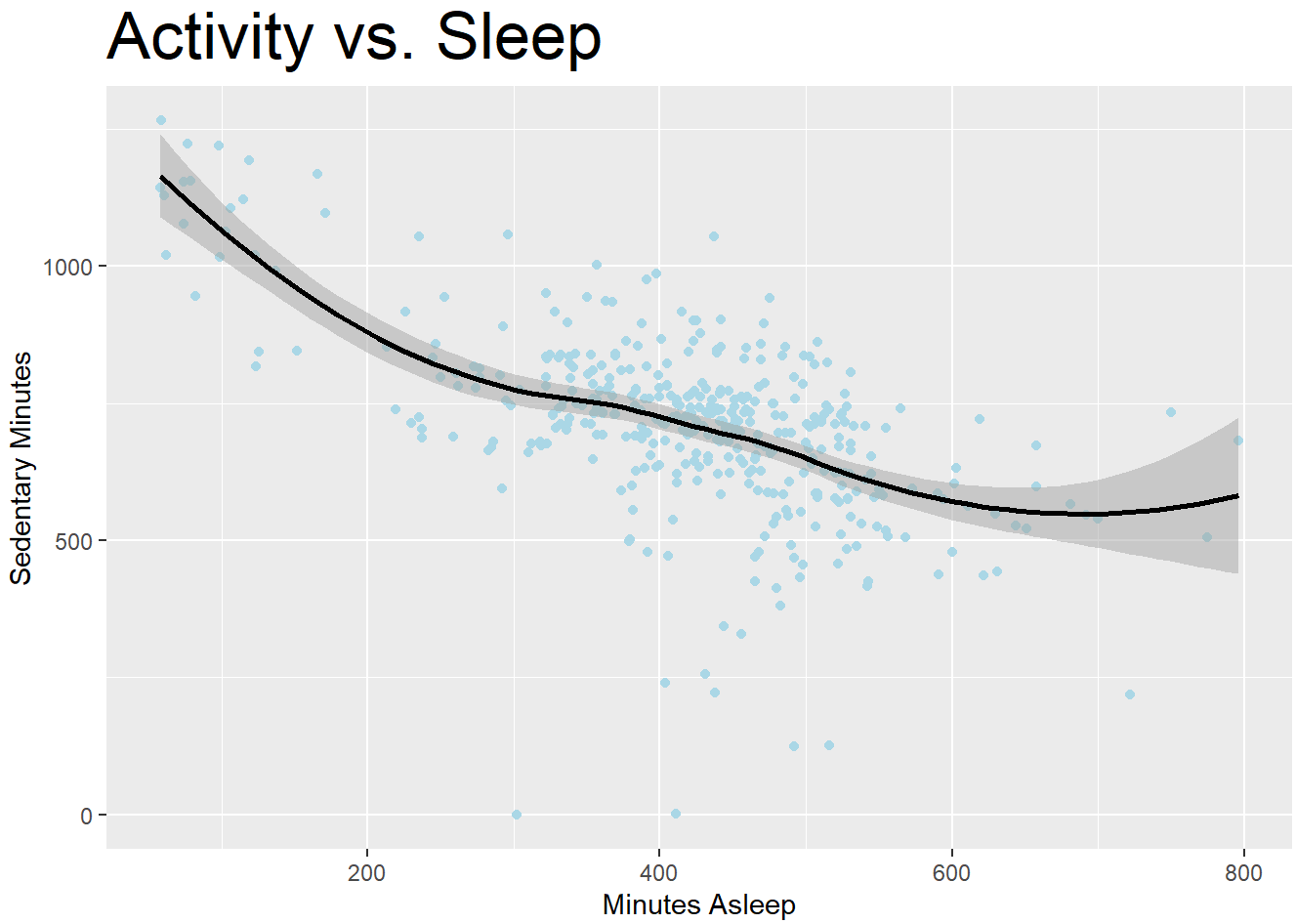
```
##           Id      Date TotalSleepRecords TotalMinutesAsleep TotalTimeInBed
## 1 1503960366 4/12/2016                 1                327            346
## 2 1503960366 4/13/2016                 2                384            407
## 3 1503960366 4/15/2016                 1                412            442
## 4 1503960366 4/16/2016                 2                340            367
## 5 1503960366 4/17/2016                 1                700            712
## 6 1503960366 4/19/2016                 1                304            320
##   TotalSteps TotalDistance TrackerDistance LoggedActivitiesDistance
## 1      13162          8.50            8.50                       0
## 2      10735          6.97            6.97                       0
## 3       9762          6.28            6.28                       0
## 4      12669          8.16            8.16                       0
## 5       9705          6.48            6.48                       0
## 6      15506          9.88            9.88                       0
##   VeryActiveDistance ModeratelyActiveDistance LightActiveDistance
## 1               1.88                     0.55                6.06
## 2               1.57                     0.69                4.71
## 3               2.14                     1.26                2.83
## 4               2.71                     0.41                5.04
## 5               3.19                     0.78                2.51
## 6               3.53                     1.32                5.03
##   SedentaryActiveDistance VeryActiveMinutes FairlyActiveMinutes
## 1                       0                25                  13
## 2                       0                21                  19
## 3                       0                29                  34
## 4                       0                36                  10
## 5                       0                38                  20
## 6                       0                50                  31
##   LightlyActiveMinutes SedentaryMinutes Calories
## 1                  328              728     1985
## 2                  217              776     1797
## 3                  209              726     1745
## 4                  221              773     1863
## 5                  164              539     1728
## 6                  264              775     2035
```

Activity vs. Sleep

I created a visualization in order to see the relationship between sleep and intense activity.

```
ggplot(data=merged_daily, aes(x=TotalMinutesAsleep, y=SedentaryMinutes))+
  geom_point(color="lightblue")+
  geom_smooth(color="black")+
  labs(title = "Activity vs. Sleep",
       y="Sedentary Minutes",
       x="Minutes Asleep")+
  theme(plot.title = element_text(size = 25))
```

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



Here we can see that users with more sedentary time have less sleep time. Therefore we can say that if a user is trying to sleep more they should be more active during the day.

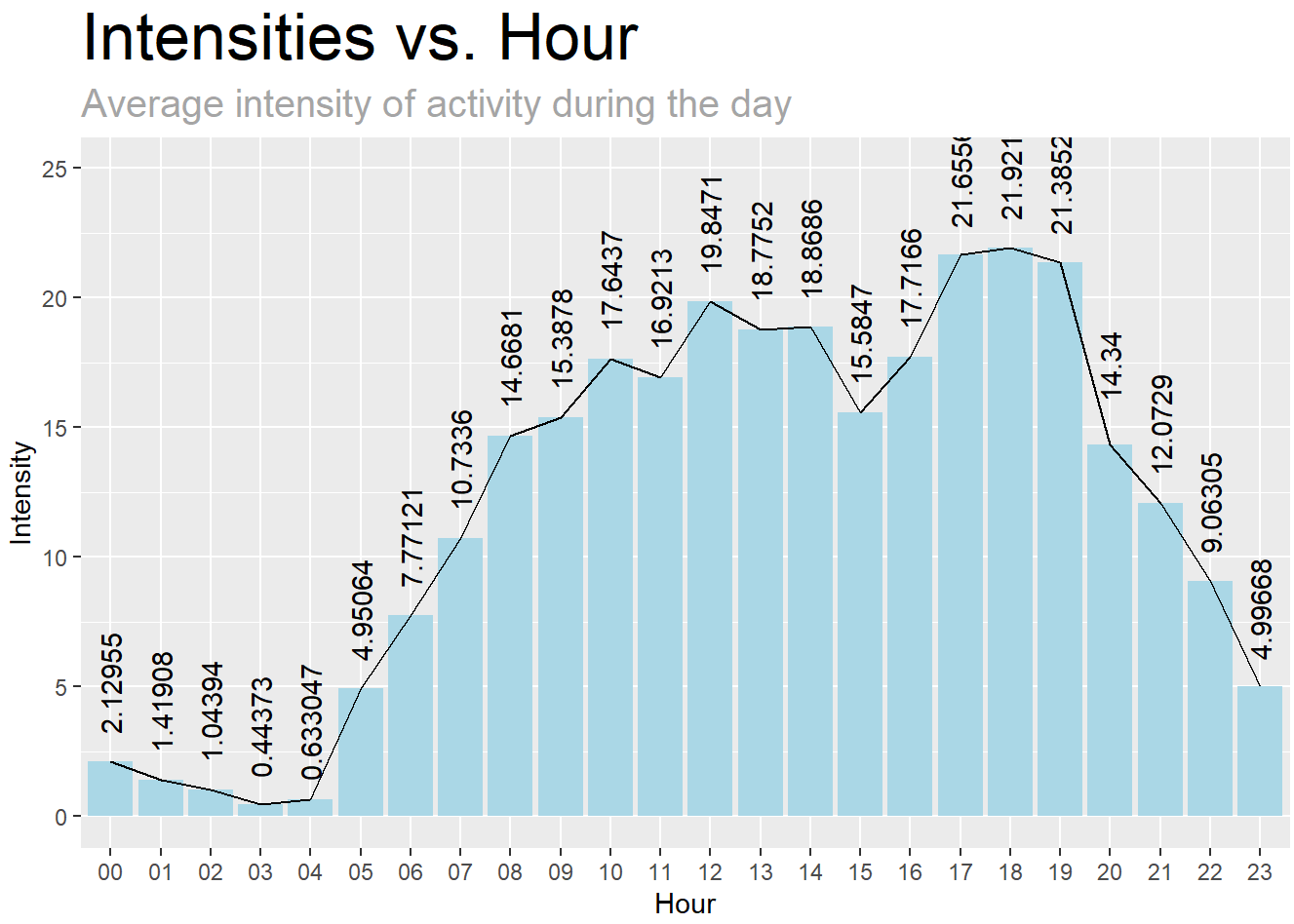
Active time

We could see at what point of the day the users are more active, then we could show them reminders around that time. Suggest to them they can be more active if they want to help improve their sleep patterns. For that, first I save a variable with the average intensity group by hour.

```
intensity_new <- hourlyIntensities %>%
  group_by(Time) %>%
  drop_na() %>%
  summarise(mean_totalIntensities = mean(TotalIntensity))
```

Now I can create a visualization with the data.

```
ggplot(data = intensity_new, aes(x=Time, y=mean_totalIntensities, group=1))+
  geom_col(fill="lightblue")+
  geom_line(colour="black", position = position_dodge(width = 1))+
  labs(title = "Intensities vs. Hour",
       subtitle = "Average intensity of activity during the day",
       y="Intensity",
       x="Hour")+
  geom_text(aes(label = signif(mean_totalIntensities)), nudge_y = 3, angle=90)+
  theme(legend.position="none", panel.spacing.x = unit(0, "npc"),
        plot.title = element_text(size = 25),
        plot.subtitle = element_text(size = 15, colour = "darkgrey"))
```



In this graph we can see that users are more active between 12 and 14, and then again between 17 and 19. We can say that those time periods are when most people exercise. This tells us that it will be better to remind users to exercise, or be active in some way, between those hours.

Steps per day

According to doctors and specialists an adult should take on average 10.000 steps a day, less than 5.000 a day becomes a sedentary lifestyle that can have a bad effect on the person's health. It can depend by sex, age, height and weight, but given this data set doesn't have that data we'll use the general numbers. The marketing team could use the data of the user to recommend the appropriate amounts of steps for that person specifically.

(Source:https://www.medicalnewstoday.com/articles/how-many-steps-should-you-take-a-day#for-physical-fitness (https://www.medicalnewstoday.com/articles/how-many-steps-should-you-take-a-day#for-physical-fitness))

We can create a visualization to see what percentage of the users in this data set take the recommended amounts of steps, or if their lifestyle is considered sedentary.

To do that I have to see how many users fulfill those amounts of steps per day and save it in a variable to use for the chart. We'll use a function to convert the sums into percent and save everything in a dataframe to use later for the chart.

```
#function to format percent number
percent <- function(num, digits = 2, ...) {
  percentage <-formatC(num, format = "f", digits = digits, ...)
  paste0(percentage, "%")
}

#Calculate the quantities
more <- sum(dailySteps$StepTotal>10000)
middle <- sum(dailySteps$StepTotal<10000 & dailySteps$StepTotal>5000)
less <- sum(dailySteps$StepTotal<5000)
total <- sum(more, middle, less)

#Create dataframe
percentage_steps <- data.frame(Steps = c(">10.000", "10.000-5.000", "<5.000" ),
                               Quantity = c(more, middle, less),
                               Percentage = percent(c(more*100/total,middle*100/total,less*100/t
otal))
                               )

head(percentage_steps)
```

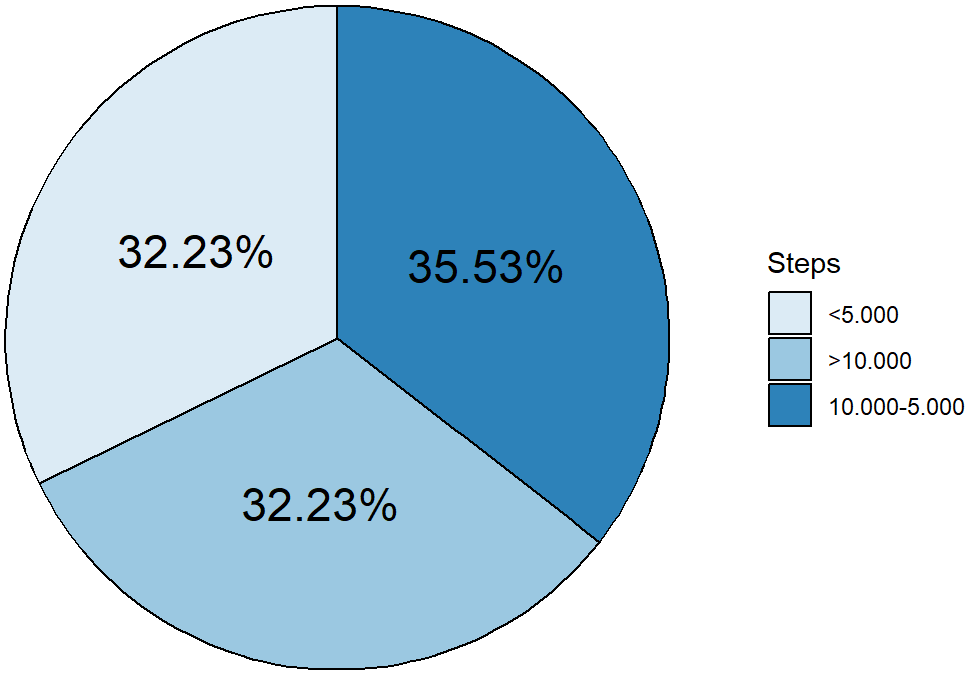
```
##          Steps Quantity Percentage
## 1      >10.000      303     32.23%
## 2 10.000-5.000      334     35.53%
## 3       <5.000      303     32.23%
```

Then I created a pie chart to show the results.

```
ggplot(data = percentage_steps, aes(x="", y=Quantity, fill=Steps))+
  geom_col(color = "black") +
  geom_text(aes(label = Percentage),
            position = position_stack(vjust = 0.5),
            size=6) +
  coord_polar("y", start=0) +
  labs(title="Goal of steps per day",
       subtitle = "Percentage of people that achieve a certain amount of steps per day")+
  theme_void()+
  theme(plot.title = element_text(size = 25),
        plot.subtitle = element_text(size = 15, colour = "darkgrey"))+
  scale_fill_brewer()
```

Goal of steps per day

Percentage of people that achieve a certain amount of steps per day



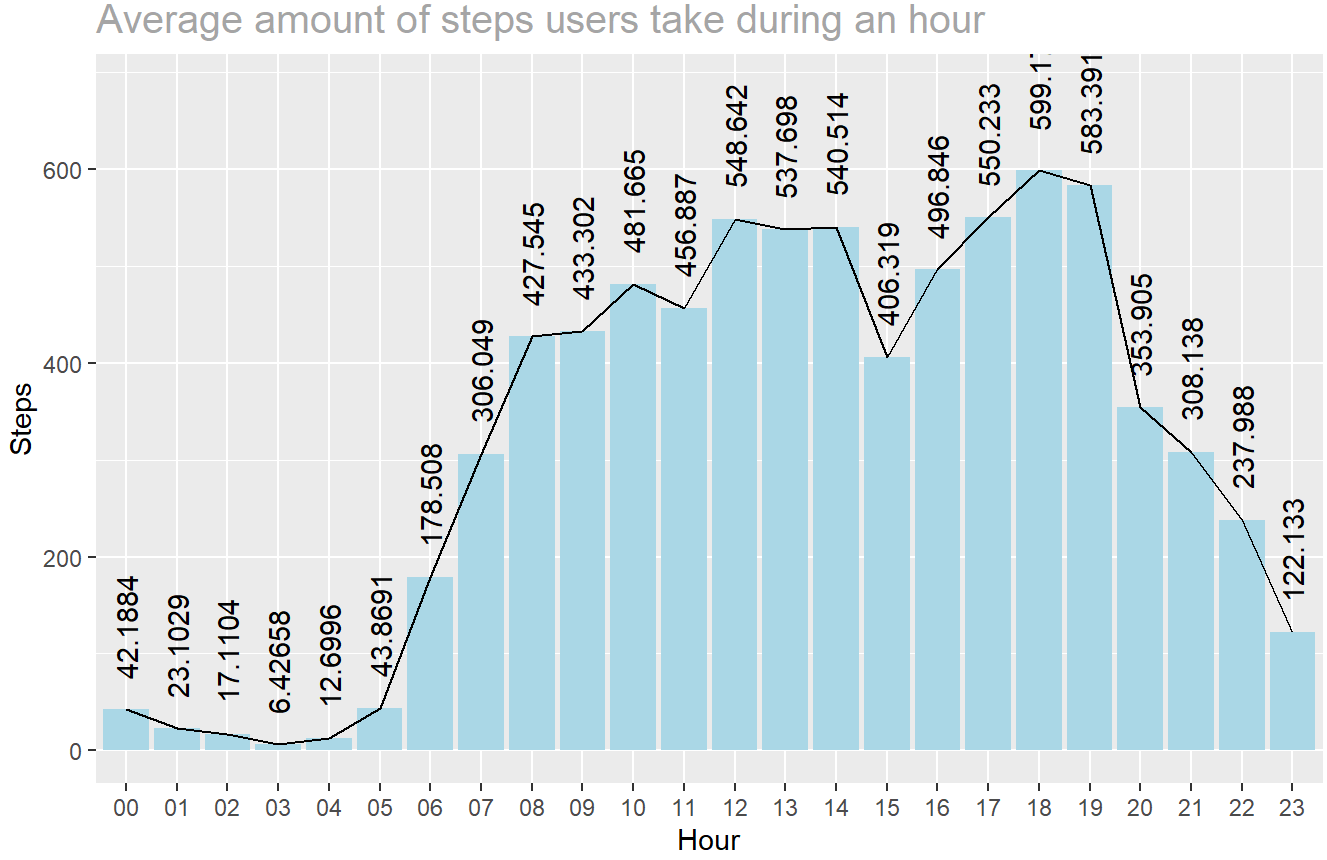
We can see in the graph that more than half of the users could improve their health if they increase their daily steps. We can use this data for the app interface with the user, for example congratulate users that achieve the average step necessarily in a day, or remind users that don't achieve it that they could increase their daily steps to improve their health according to medical professionals recommendations.

Step during the day

I created a graph that will show when the user takes more steps, we can use that data to know when to push the reminders in the app. For that graph first I needed to save a variable with the average steps group by hour.

```
steps_new <- hourlySteps %>%
  group_by(Time) %>%
  drop_na() %>%
  summarise(mean_StepTotal = mean(StepTotal))

ggplot(data = steps_new, aes(x=Time, y=mean_StepTotal, group=1))+
  geom_col(fill="lightblue")+
  geom_line(colour="black", position = position_dodge(width = 1))+
  labs(title = "Steps vs. Hour",
       subtitle = "Average amount of steps users take during an hour",
       y="Steps",
       x="Hour")+
  geom_text(aes(label = signif(mean_StepTotal)), nudge_y = 85, angle=90)+
  theme(legend.position="none", panel.spacing.x = unit(0, "npc"),
        plot.title = element_text(size = 25),
        plot.subtitle = element_text(size = 15, colour = "darkgrey"))
```



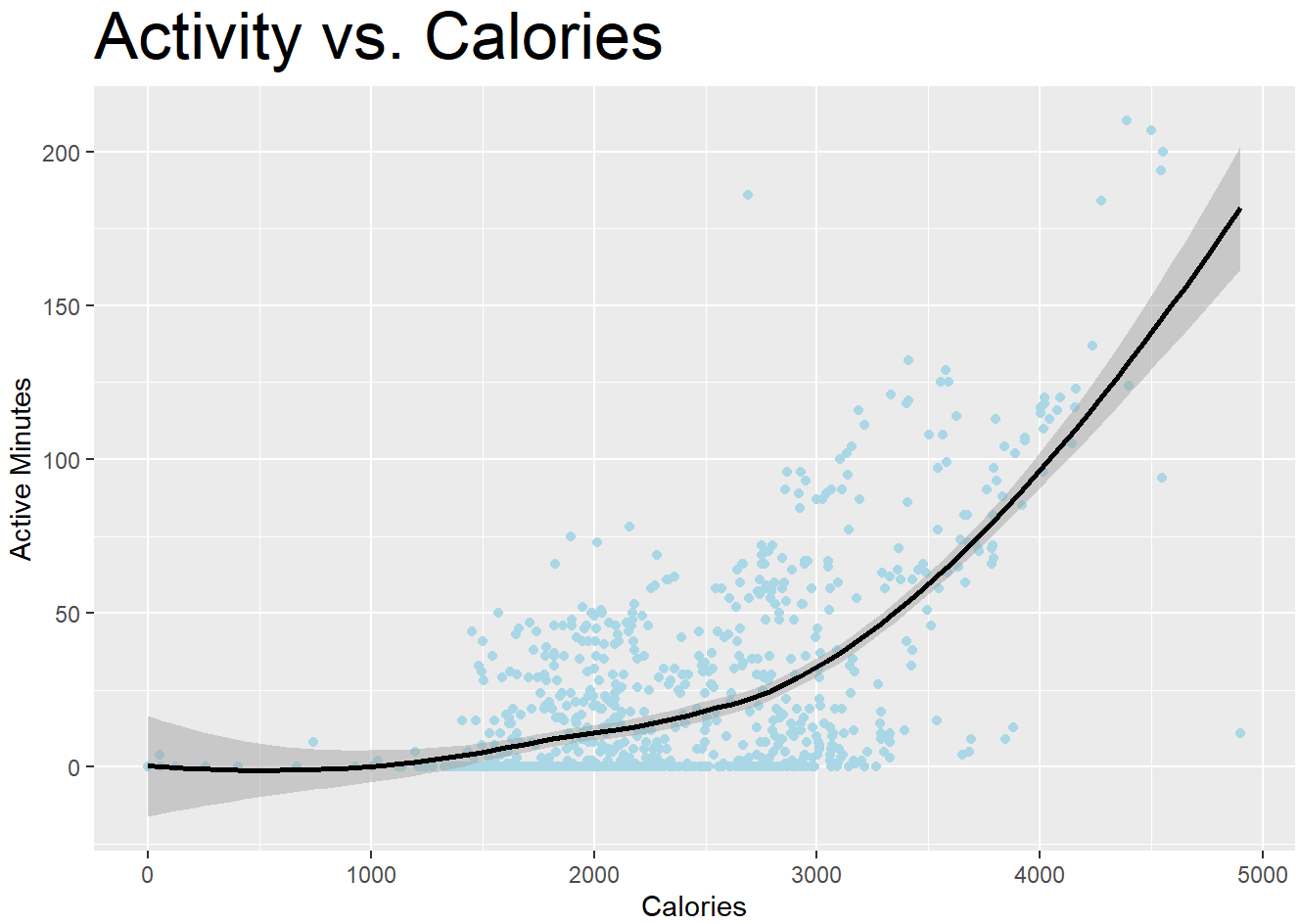
Seeing the chart we can say that the best time for the reminders will be from 12 to 14 and from 17 to 19, that is the time users take the most steps.

Activity vs. Calories

I created a chart to see if there is a relationship between calories and activity.

```
ggplot(data=dailyActivity, aes(x=Calories, y=VeryActiveMinutes))+
  geom_point(color="lightblue")+
  geom_smooth(color="black")+
  labs(title = "Activity vs. Calories",
       y="Active Minutes")+
  theme(plot.title = element_text(size = 25))
```

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



In the chart we can see a positive relationship between activity and calories, the more activity the user does the more calories are being used.

With this we can determine that when a user is active more calories are being used, therefore it's likely that a user of the app for the purpose of weight loss will like to keep track of their caloric intake as well as used calories during the day. We can add that feature to attract more users.

Calories used during the day

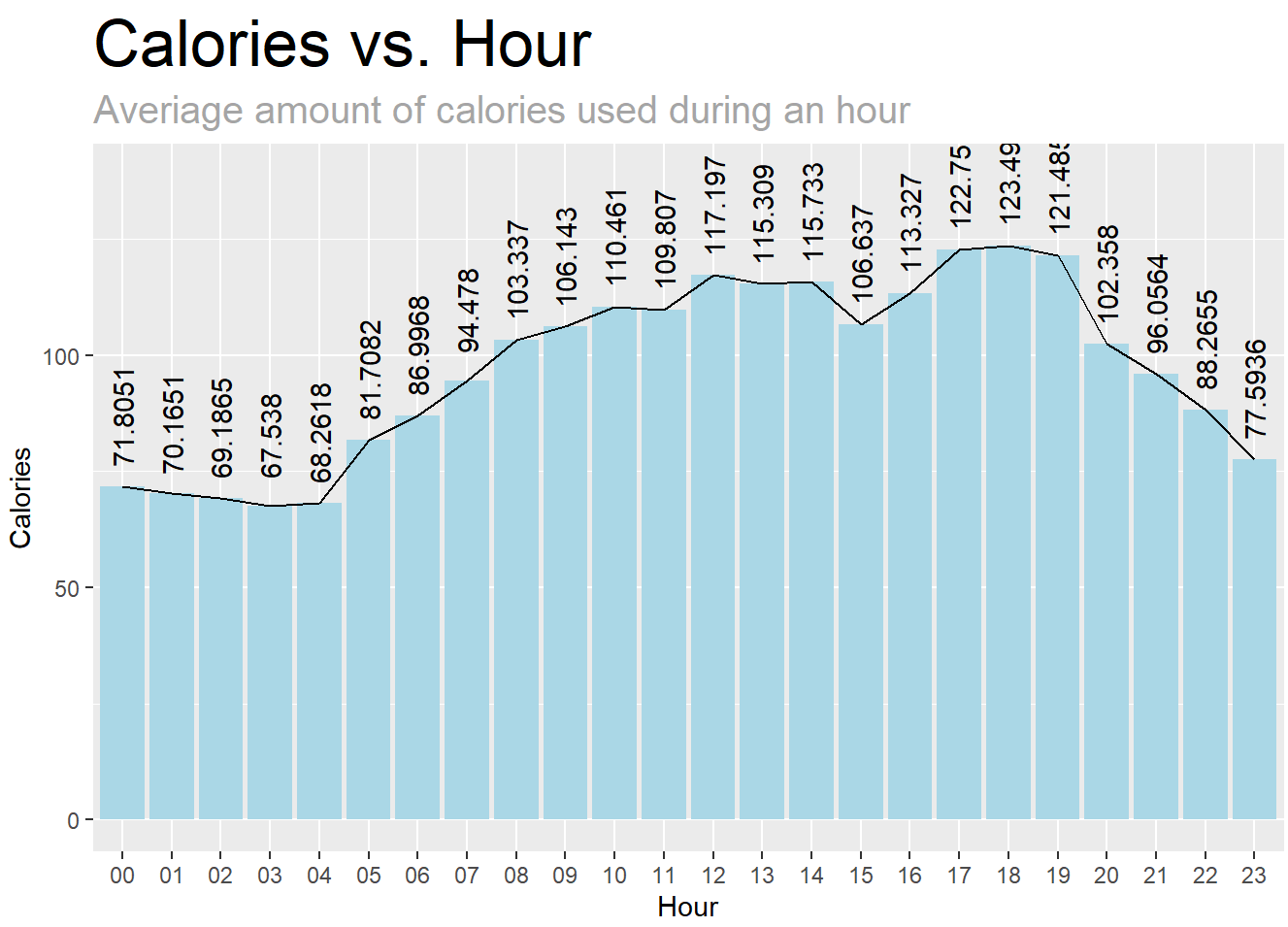
Finally we can use the data on calories used daily and hourly to make suggestions to the user with recipes for snacks or food that are healthy and obviously delicious as well. Doing this in a moment that the user uses a big amount of calories will be best since they are more likely to be hungry and looking for something to make.

To determine when to suggest recipes to the user, depending on the level of activity and calories used, I created a chart with the used calories hourly.

To do this I needed to save a new variable with the average calories used per hour.

```
calories_new <- hourlyCalories %>%
  group_by(Time) %>%
  drop_na() %>%
  summarise(mean_totalCalories = mean(Calories))

ggplot(data = calories_new, aes(x=Time, y=mean_totalCalories, group=1))+
  geom_col(fill="lightblue")+
  geom_line(colour="black", position = position_dodge(width = 1))+
  labs(title = "Calories vs. Hour",
       subtitle = "Averiage amount of calories used during an hour",
       y = "Calories",
       x="Hour")+
  geom_text(aes(label = signif(mean_totalCalories)), nudge_y = 15, angle=90)+
  theme(legend.position="none", panel.spacing.x = unit(0, "npc"),
        plot.title = element_text(size = 25),
        plot.subtitle = element_text(size = 15, colour = "darkgrey"))
```



In the chart we see two peaks in the calories used, between 12 and 14 and also between 17 and 19. Those time frames will be better to push recipes and tips for healthy eating to users.

Conclusions

I'll focus on the Bellabeat Membership. After what the analysis showed I can make the following recommendations to the bellabeat team:

- They should focus their marketing to people that are interested in improving their health, maybe for a better life quality.
- They should add a feature that allows users to keep track of their caloric intake. It would be useful for example for people that use the app to try to lose weight or are in a regimen for mass and muscle building.
- They should market the sleep and healthy weight log features because those are the ones people use less but they could benefit from them, so it might mean that the users are not aware or don’t know how to use them.
- They could suggest recipes for healthy meals or snacks around the periods of times when people are more active which will make them hungry and they will most likely be looking for something to cook.
- They could use a reward system or suggestions for people about their daily steps based on the suggested amount by medical professionals.
- They could suggest to them they can be more active if they want to help improve their sleep patterns and they can track it using a sleep log feature in the app.
- All of these suggestions will be better offered around the times from 12:00 to 14:00 and 17:00 and 19:00, because that’s where people are more active and take the most steps.