Completed by: Fardin Islam Mahin (https://www.linkedin.com/in/fardin-islam-mahin/)

The Business Statement

I have to examine FitBit fitness tracker user data to understand usage patterns and trends for Bellabeat's marketing strategy. The business objectives include identifying trends in the smart device usage, determining their relevance to Bellabeat's customers, and using these trends to influence Bellabeat's marketing strategy. The primary stakeholders in this project are Urška Sršen (Co-founder and Chief Creative Officer), Sando Mur (Mathematician and Co-founder), and the Bellabeat Marketing Analytics Team, who are responsible for gathering and analyzing data to inform the company's marketing efforts. My analysis aims to provide insights for tailored marketing strategies and pointing out the aspects that competitors may not be paying attention to.

Preparation

The FitBit Fitness Tracker Data (https://www.kaggle.com/arashnic/fitbit), available on Kaggle and made accessible through *Mobius* (https://www.kaggle.com/arashnic) contains information from thirty Fitbit users, offering minute-level details on physical activity, heart rate, and sleep monitoring, along with daily activity and step counts in 18 separate CSV files. While this dataset is valuable for the analysis, it has some limitations. I am doing some ROCCC Evaluation to summarize those:

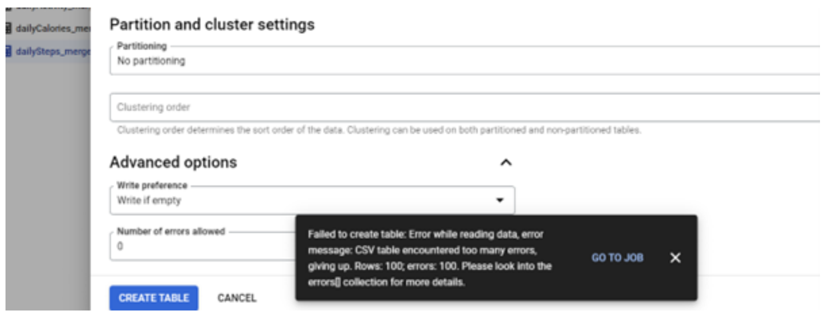
- Reliable: The reliability of this dataset is considered low due to the limited sample size of 30 users, potentially having sampling bias.
- Original: The data is sourced from a third party.
- Comprehensive: It may provide valuable insights that aligns with the parameters of the business task.
- Current: The data is relatively outdated as it is collected in 2016 and may not accurately represent current user habits.
- Cited: The dataset's sources are unspecified which indicates low credibility.

I will be using Google's BigQuery and somewhat MS Excel to clean, R Programming Language to analyze and vizualize the data in:

- dailyActivity_merged.csv
- sleepDay_merged.csv
- weightLogInfo_merged.csv

Cleaning

Loading the CSV files in BigQuery Console project. While loading sleepDay_merged.csv in Bigquery, the error message:



Partition and cluster settings

Partitioning: No partitioning

Clustering order

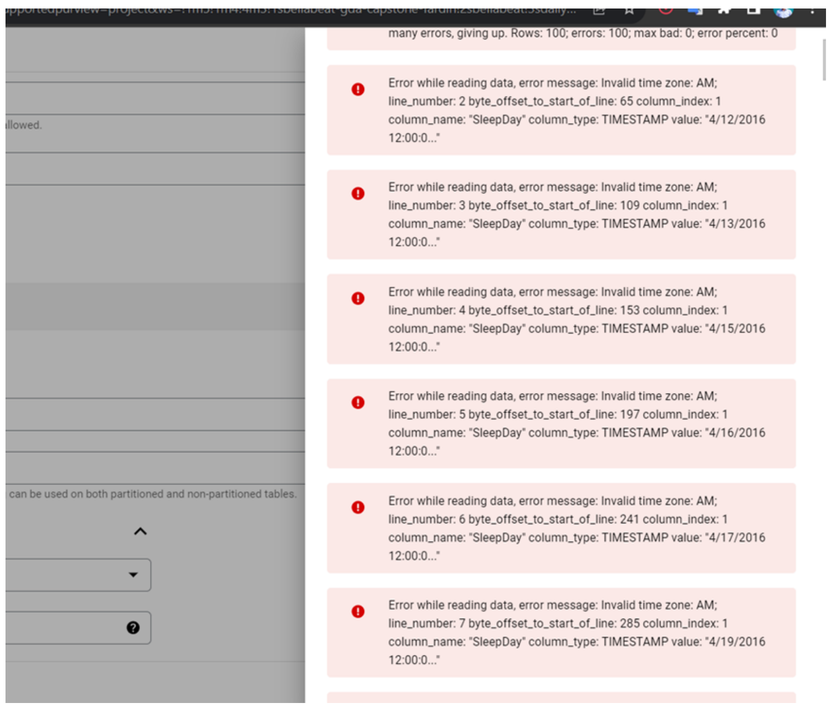
Clustering order determines the sort order of the data. Clustering can be used on both partitioned and non-partitioned tables.

Advanced options

Write preference: Write if empty

Number of errors allowed: 0

```
Failed to create table: Error while reading data, error message: CSV table encountered too many errors, giving up. Rows: 100; errors: 100. Please look into the errors[] collection for more details.
```

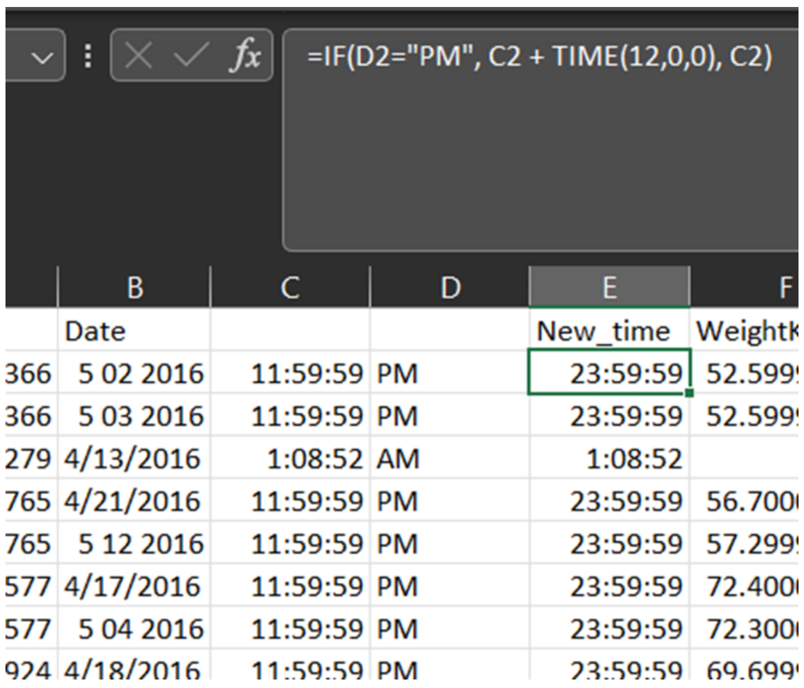


```
many errors, giving up. Rows: 100; errors: 100; max bad: 0; error percent: 0

Error while reading data, error message: Invalid time zone: AM; line_number: 2 byte_offset_to_start_of_line: 65 column_index: 1 column_name: "SleepDay" column_type: TIMESTAMP value: "4/12/2016 12:00:0..."

Error while reading data, error message: Invalid time zone: AM; line_number: 3 byte_offset_to_start_of_line: 109 column_index: 1 column_name: "SleepDay" column_type: TIMESTAMP value: "4/13/2016 12:00:0..."

Error while reading data, error message: Invalid time zone: AM; line_number: 4 byte_offset_to_start_of_line: 153 column_index: 1 column_name: "SleepDay" column_type: TIMESTAMP value: "4/15/2016 12:00:0..."

Error while reading data, error message: Invalid time zone: AM; line_number: 5 byte_offset_to_start_of_line: 197 column_index: 1 column_name: "SleepDay" column_type: TIMESTAMP value: "4/16/2016 12:00:0..."

Error while reading data, error message: Invalid time zone: AM; line_number: 6 byte_offset_to_start_of_line: 241 column_index: 1 column_name: "SleepDay" column_type: TIMESTAMP value: "4/17/2016 12:00:0..."

Error while reading data, error message: Invalid time zone: AM; line_number: 7 byte_offset_to_start_of_line: 285 column_index: 1 column_name: "SleepDay" column_type: TIMESTAMP value: "4/19/2016 12:00:0..."
```

So needed to delete " AM" from the SleepDay column's data. Did it in **MS Excel** using =SUBSTITUTE(B2, " AM", "") formula.

Same issue is for Date column of weightLogInfo_merged.csv data:

First separated the data of date column in 3 columns using Text to column feature. Then used =IF(D2="PM", C2 + TIME(12,0,0), C2) in a new to make the time consistent. Then removed unnecessary columns.

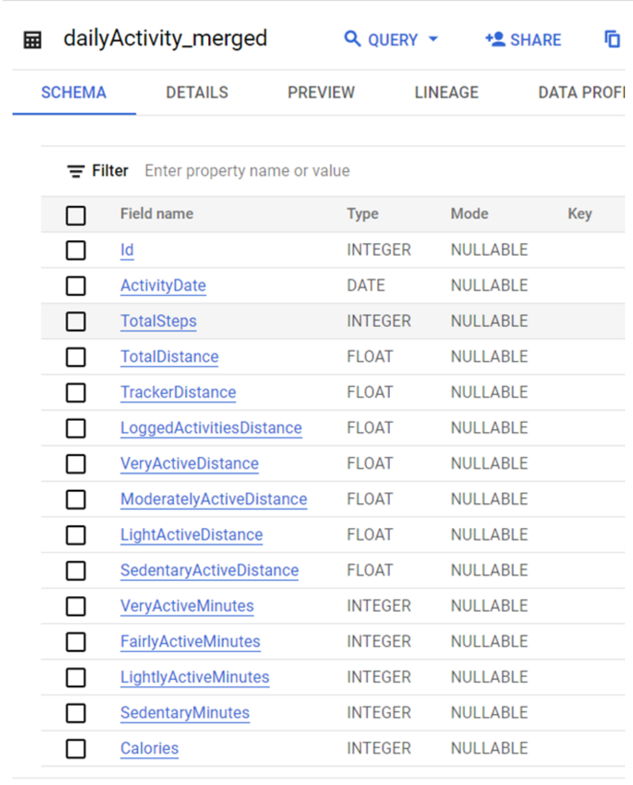


=IF(D2="PM", C2 + TIME(12,0,0), C2)

	B	C	D	E	F
	Date			New_time	WeightK
366	5 02 2016	11:59:59	PM	23:59:59	52.599
366	5 03 2016	11:59:59	PM	23:59:59	52.599
279	4/13/2016	1:08:52	AM	1:08:52	
765	4/21/2016	11:59:59	PM	23:59:59	56.700
765	5 12 2016	11:59:59	PM	23:59:59	57.299
577	4/17/2016	11:59:59	PM	23:59:59	72.400
577	5 04 2016	11:59:59	PM	23:59:59	72.300
924	4/18/2016	11:59:59	PM	23:59:59	69.699

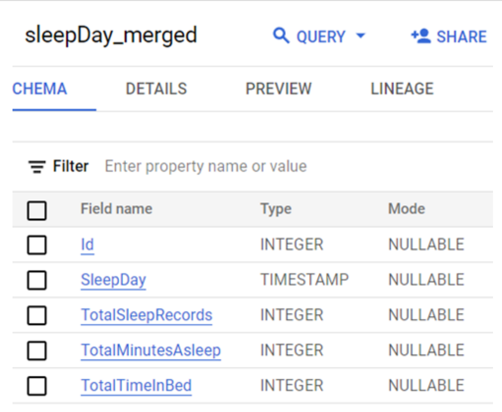
Finally all the 3 tables are loaded!

Schema of the tables after loading:



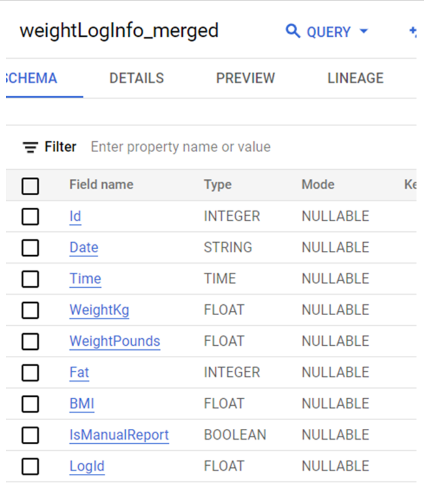
dailyActivity_merged

Field name	Type	Mode	Key
Id	INTEGER	NULLABLE	
ActivityDate	DATE	NULLABLE	
TotalSteps	INTEGER	NULLABLE	
TotalDistance	FLOAT	NULLABLE	
TrackerDistance	FLOAT	NULLABLE	
LoggedActivitiesDistance	FLOAT	NULLABLE	
VeryActiveDistance	FLOAT	NULLABLE	
ModeratelyActiveDistance	FLOAT	NULLABLE	
LightActiveDistance	FLOAT	NULLABLE	
SedentaryActiveDistance	FLOAT	NULLABLE	
VeryActiveMinutes	INTEGER	NULLABLE	
FairlyActiveMinutes	INTEGER	NULLABLE	
LightlyActiveMinutes	INTEGER	NULLABLE	
SedentaryMinutes	INTEGER	NULLABLE	
Calories	INTEGER	NULLABLE	



sleepDay_merged

Field name	Type	Mode
Id	INTEGER	NULLABLE
SleepDay	TIMESTAMP	NULLABLE
TotalSleepRecords	INTEGER	NULLABLE
TotalMinutesAsleep	INTEGER	NULLABLE
TotalTimeInBed	INTEGER	NULLABLE



weightLogInfo_merged

Field name	Type	Mode
Id	INTEGER	NULLABLE
Date	STRING	NULLABLE
Time	TIME	NULLABLE
WeightKg	FLOAT	NULLABLE
WeightPounds	FLOAT	NULLABLE
Fat	INTEGER	NULLABLE
BMI	FLOAT	NULLABLE
IsManualReport	BOOLEAN	NULLABLE
LogId	FLOAT	NULLABLE

Use of BigQuery for Formatting, Handling Duplicates and Missing Data and Joining tables.

Find the full code here (https://github.com/fardiiin/Bellabeat-GDA-Capstone-SQL-R/blob/main/bigquery.sql).

Finally, the tables are exported as CSV files (can be found in my Github repository (https://github.com/fardiiin/Bellabeat-GDA-Capstone-SQL-R)). Then loaded in positCloud project for analysis with R programming language.

Analysis and Vizualization

Using R Programming language, I will proceed from here on.

Average daily steps, distance covered and active minutes by user

```r
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✗ dplyr::filter() masks stats::filter()
## ✗ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
average_data <- read_csv("dailyActivity.csv") %>%
  group_by(Id) %>%
  summarise(
    AverageDailySteps = mean(TotalSteps),
    AverageDailyDistance = mean(TotalDistance),
    AverageActiveMinutes = mean(VeryActiveMinutes + FairlyActiveMinutes + LightlyActiveMinutes)
  ) %>%
  arrange(Id)
```

```
## Rows: 940 Columns: 15
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl  (14): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesD...
## date  (1): ActivityDate
##
## 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.
```

```
print(average_data)
```

```
## # A tibble: 33 × 4
##            Id AverageDailySteps AverageDailyDistance AverageActiveMinutes
##         <dbl>             <dbl>                <dbl>                <dbl>
##  1 1503960366            12117.                7.81                 278.
##  2 1624580081             5744.                3.91                 168.
##  3 1644430081             7283.                5.30                 209.
##  4 1844505072             2580.                1.71                 117.
##  5 1927972279              916.                0.635                 40.7
##  6 2022484408            11371.                8.08                 313.
##  7 2026352035             5567.                3.45                 257
##  8 2320127002             4717.                3.19                 202.
##  9 2347167796             9520.                6.36                 287.
## 10 2873212765             7556.                5.10                 328.
## # ℹ 23 more rows
```

Average sleep duration by user

```
average_sleep_duration <- read_csv("dailyActivitySleep.csv") %>%
  group_by(Id) %>%
  summarize(AvgSleepMin = mean(TotalMinutesAsleep)) %>%
    arrange(Id)
```

```
## Rows: 410 Columns: 18
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl  (17): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesD...
## date  (1): ActivityDate
##
## 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.
```

```
print(average_sleep_duration)
```

```
## # A tibble: 24 × 2
##            Id AvgSleepMin
##         <dbl>       <dbl>
##  1 1503960366        360.
##  2 1644430081        294
##  3 1844505072        652
##  4 1927972279        417
##  5 2026352035        506.
##  6 2320127002         61
##  7 2347167796        447.
##  8 3977333714        294.
##  9 4020332650        349.
## 10 4319703577        477.
## # ℹ 14 more rows
```

Average Weight and BMI for each user

```
average_weightBMI <- read_csv("weightLogInfo.csv") %>%
  group_by(Id) %>%
  summarize(AvgWeightKg = mean(WeightKg), AvgBMI = mean(BMI))
```

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

```
print(average_weightBMI)
```

```
## # A tibble: 8 × 3
##           Id AvgWeightKg AvgBMI
##        <dbl>       <dbl>  <dbl>
## 1 1503960366        52.6   22.6
## 2 1927972279       134.    47.5
## 3 2873212765        57     21.6
## 4 4319703577        72.4   27.4
## 5 4558609924        69.6   27.2
## 6 5577150313        90.7   28
## 7 6962181067        61.6   24.0
## 8 8877689391        85.1   25.5
```

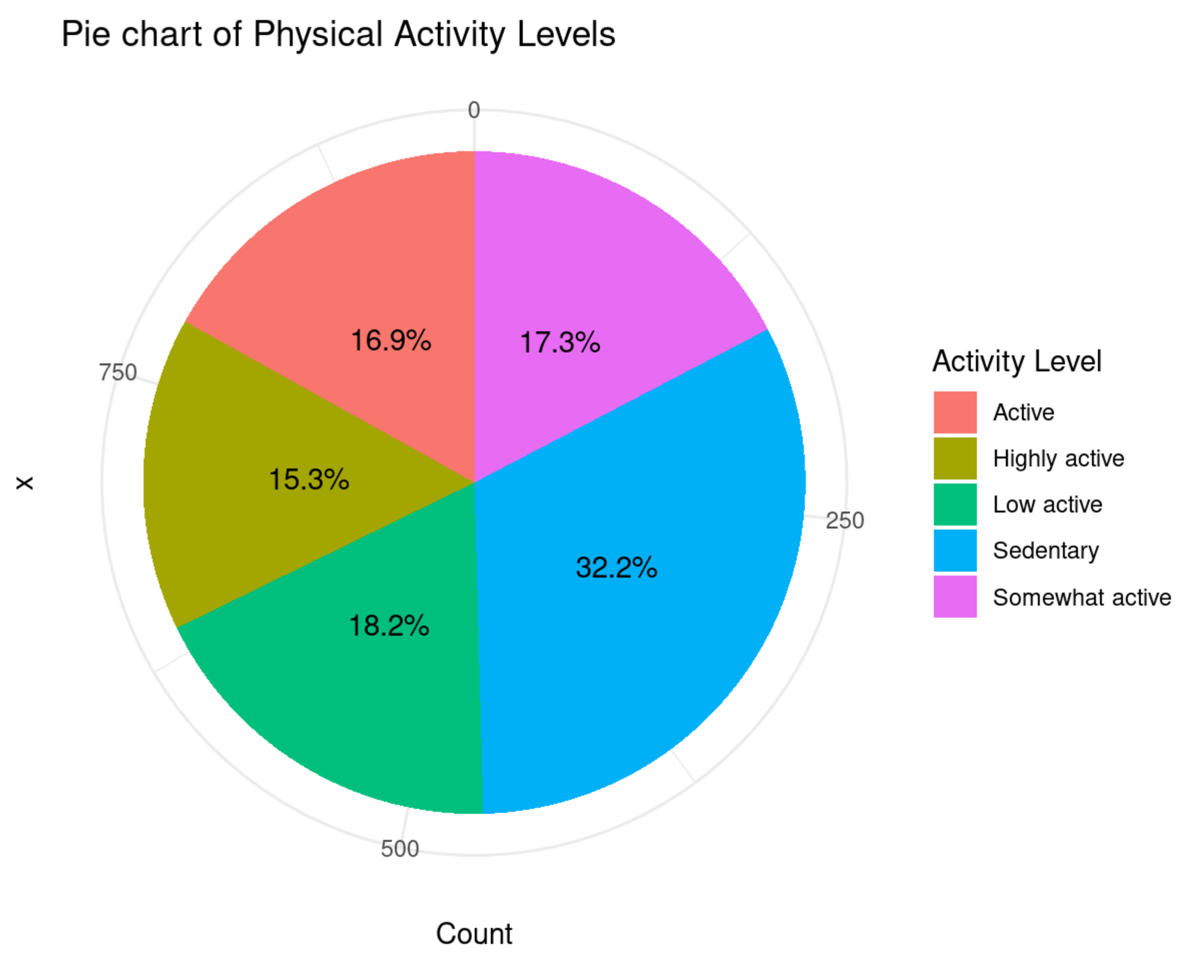
Physical activity levels

- Sedentary (< 5,000 steps/day): Individuals who take fewer than 5,000 steps per day are categorized as sedentary. This level of activity is considered insufficient for maintaining good health and fitness.
- Low Active (5,000-7,499 steps/day): People falling in this category are considered “low active.” While they are more active than sedentary individuals, they still have room for improvement in terms of physical activity.
- Somewhat Active (7,500-9,999 steps/day): Individuals taking between 7,500 and 9,999 steps per day are classified as “somewhat active.” This level of activity suggests a more moderate commitment to physical fitness.
- Active (≥10,000-12,499 steps/day): Those who achieve between 10,000 and 12,499 steps per day are labeled as “active.” This level of activity is associated with better health and fitness.
- Highly Active (≥12,500 steps/day): Finally, individuals who take 12,500 steps or more per day are considered “highly active.” They are at the upper end of the spectrum in terms of physical activity and are likely to experience the most health benefits from their active lifestyle.

```
activity_data <- read_csv("dailyActivity.csv") %>%
  mutate(ActivityLevel = case_when(
    TotalSteps < 5000 ~ "Sedentary",
    between(TotalSteps, 5000, 7499) ~ "Low active",
    between(TotalSteps, 7500, 9999) ~ "Somewhat active",
    between(TotalSteps, 10000, 12499) ~ "Active",
    TotalSteps >= 12500 ~ "Highly active"
  ))
```

```
## Rows: 940 Columns: 15
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl  (14): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesD...
## date  (1): ActivityDate
##
## 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.
```

```
activity_summary <- activity_data %>%
  group_by(ActivityLevel) %>%
  summarize(Count = n())

activity_summary <- activity_summary %>%
  mutate(Percentage = (Count / sum(Count)) * 100)

library(ggplot2)

ggplot(activity_summary, aes(x = "", y = Count, fill = ActivityLevel, label = paste0(round(Percentage, 1), "%"))) +
  geom_bar(stat = "identity") +
  geom_text(position = position_stack(vjust = 0.5)) +
  coord_polar("y") +
  labs(title = "Pie chart of Physical Activity Levels", fill = "Activity Level")+
  theme_minimal()
```



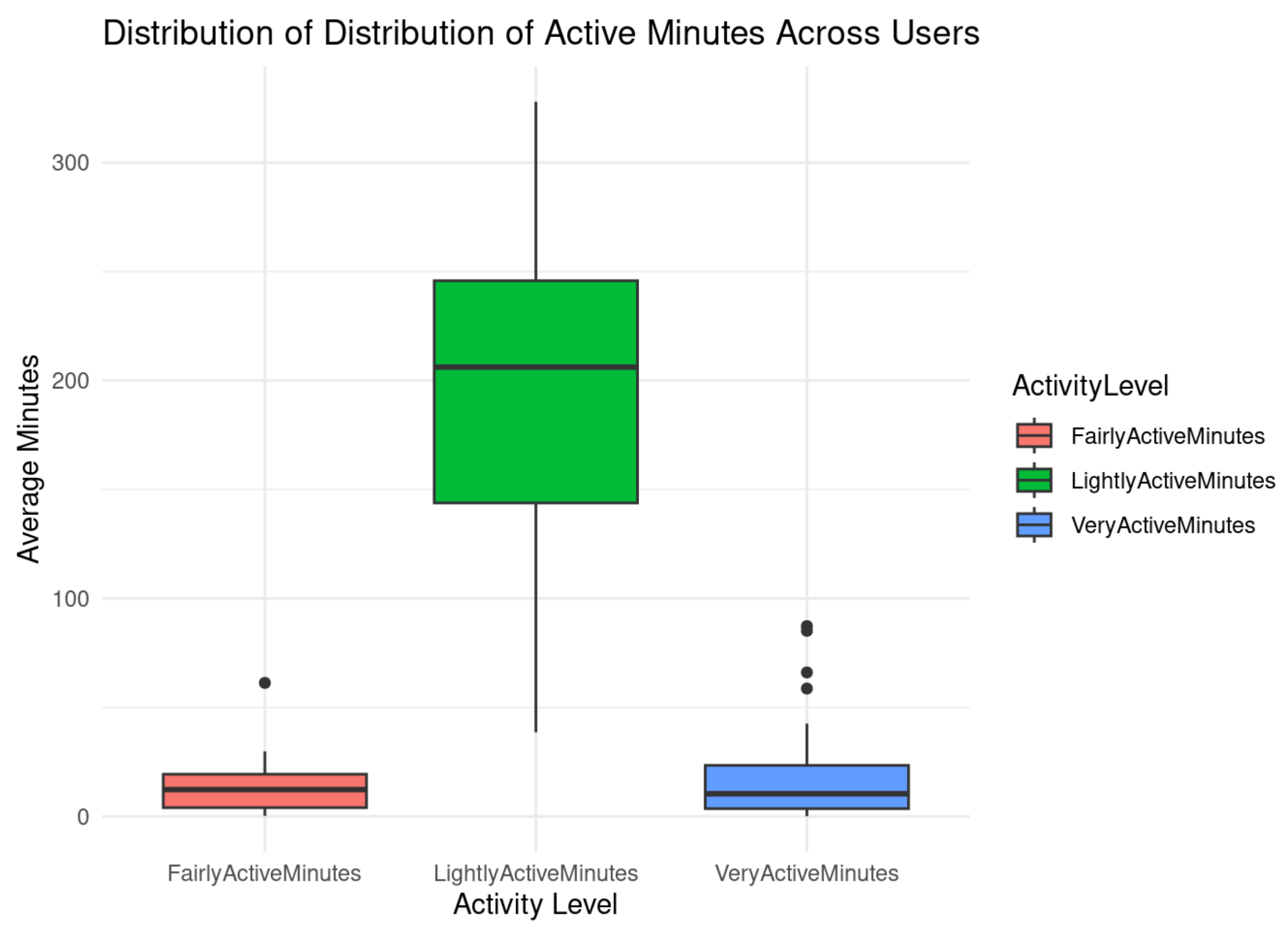
Sedentary activity percentage is high.

Distribution of Active Minutes

```
activity_distribution <- read_csv("dailyActivity.csv") %>%
  group_by(Id) %>%
  summarise(
    VeryActiveMinutes = mean(VeryActiveMinutes),
    FairlyActiveMinutes = mean(FairlyActiveMinutes),
    LightlyActiveMinutes = mean(LightlyActiveMinutes)
  )
```

```
## Rows: 940 Columns: 15
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl  (14): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesD...
## date  (1): ActivityDate
##
## 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.
```

```
activity_distribution_long <- activity_distribution %>%
  pivot_longer(
    cols = c(VeryActiveMinutes, FairlyActiveMinutes, LightlyActiveMinutes),
    names_to = "ActivityLevel",
    values_to = "AverageMinutes"
  )

ggplot(activity_distribution_long, aes(x = ActivityLevel, y = AverageMinutes, fill = ActivityLevel)) +
  geom_boxplot() +
  labs(
    title = "Distribution of Distribution of Active Minutes Across Users",
    x = "Activity Level",
    y = "Average Minutes"
  ) + theme_minimal()
```



Summary of Active days of the Week and Month

```
dailyActivity <- read_csv("dailyActivity.csv")
```

```
## Rows: 940 Columns: 15
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl  (14): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesD...
## date  (1): ActivityDate
##
## 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.
```

```
dailyActivity$DayOfWeek <- weekdays(dailyActivity$ActivityDate)
dailyActivity$Month <- format(dailyActivity$ActivityDate, "%B")

head(dailyActivity)
```

```
## # A tibble: 6 × 17
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
##        <dbl> <date>            <dbl>         <dbl>           <dbl>
## 1 1624580081 2016-05-01        36019         28.0            28.0 
## 2 1644430081 2016-04-14        11037          8.02            8.02
## 3 1644430081 2016-04-19        11256          8.18            8.18
## 4 1644430081 2016-04-28         9405          6.84            6.84
## 5 1644430081 2016-04-30        18213         13.2            13.2 
## 6 1644430081 2016-05-03        12850          9.34            9.34
## # ℹ 12 more variables: LoggedActivitiesDistance <dbl>,
## #   VeryActiveDistance <dbl>, ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, SedentaryActiveDistance <dbl>,
## #   VeryActiveMinutes <dbl>, FairlyActiveMinutes <dbl>,
## #   LightlyActiveMinutes <dbl>, SedentaryMinutes <dbl>, Calories <dbl>,
## #   DayOfWeek <chr>, Month <chr>
```

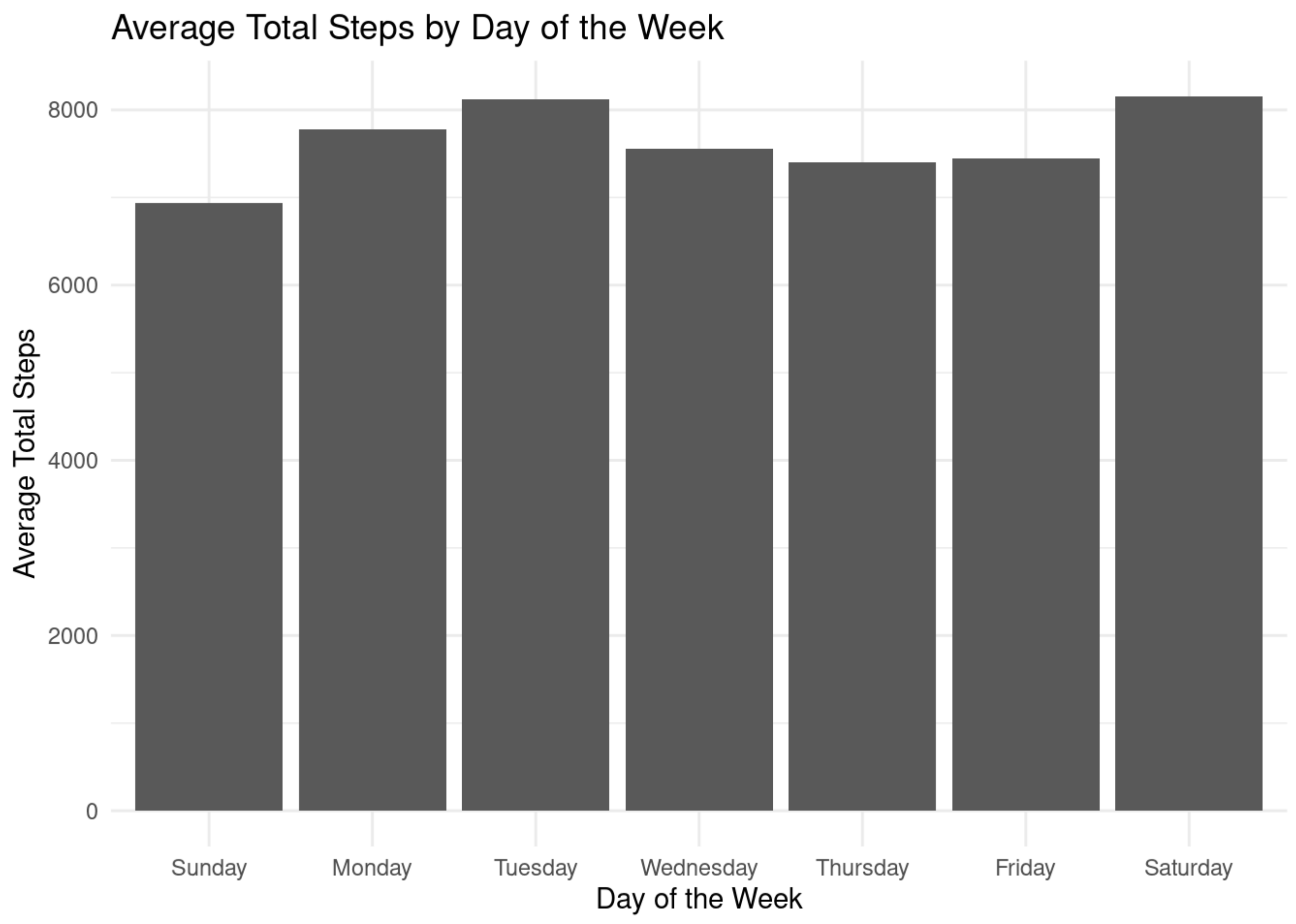
```
DayOfWeekSummary <- dailyActivity %>%
  group_by(DayOfWeek) %>%
  summarize(AvgTotalSteps = mean(TotalSteps))

DayOfWeekSummary <- DayOfWeekSummary %>%
  arrange(factor(DayOfWeek, levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday")))

print(DayOfWeekSummary)
```

```
## # A tibble: 7 × 2
##   DayOfWeek AvgTotalSteps
##   <chr>             <dbl>
## 1 Monday            7781.
## 2 Tuesday           8125.
## 3 Wednesday         7559.
## 4 Thursday          7406.
## 5 Friday            7448.
## 6 Saturday          8153.
## 7 Sunday            6933.
```

```
monthSummary <- dailyActivity %>%
  group_by(Month) %>%
  summarize(AvgTotalSteps = mean(TotalSteps))
print(monthSummary)
```

```
## # A tibble: 2 × 2
##   Month AvgTotalSteps
##   <chr>         <dbl>
## 1 April         7811.
## 2 May           7316.
```

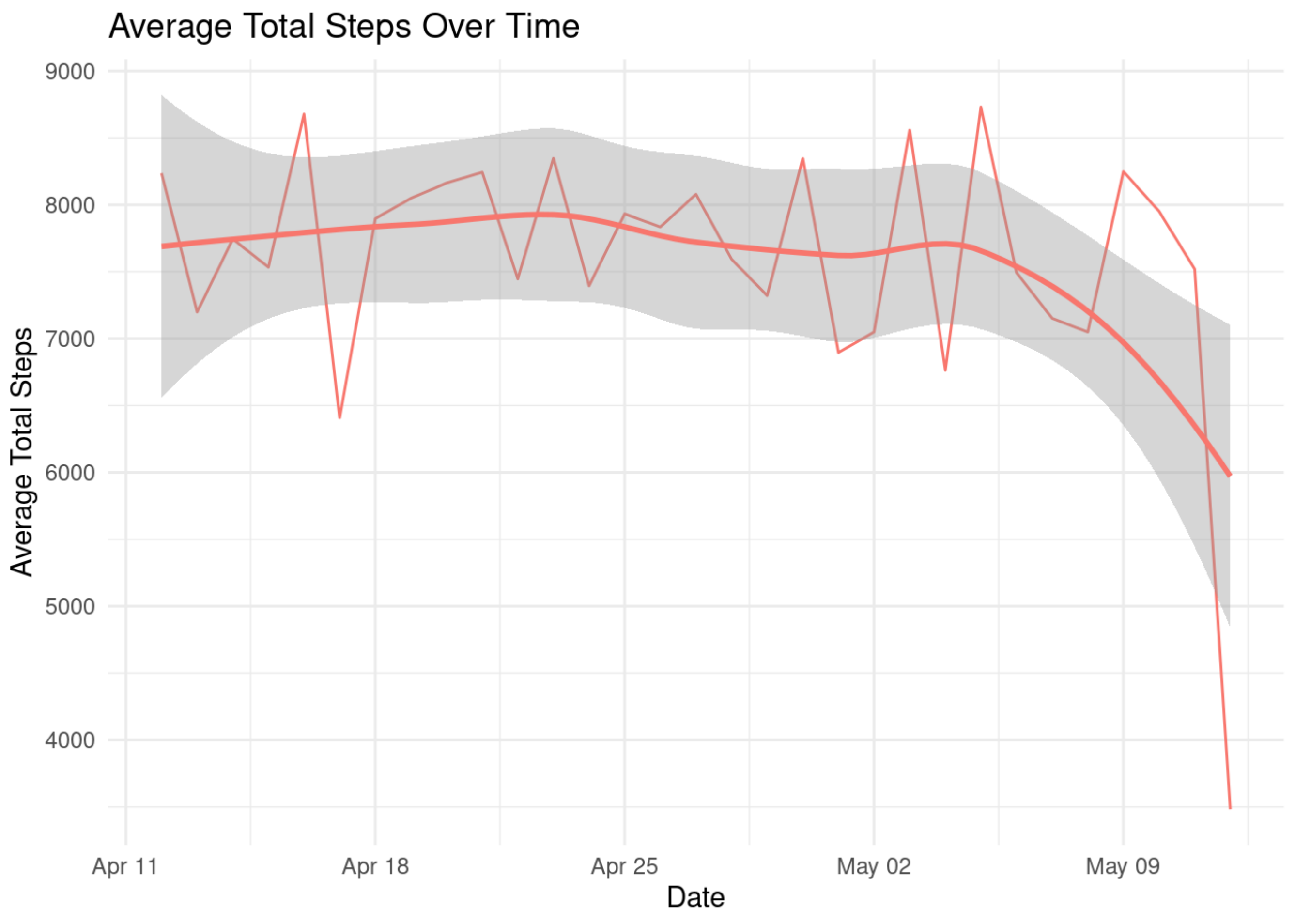
```
DayOfWeekSummary$DayOfWeek <- factor(DayOfWeekSummary$DayOfWeek,
                                      levels = c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturda
y"))

ggplot(DayOfWeekSummary, aes(x = DayOfWeek, y = AvgTotalSteps)) +
  geom_bar(stat = "identity") +
  labs(title = "Average Total Steps by Day of the Week",
       x = "Day of the Week",
       y = "Average Total Steps") + theme_minimal()
```



On Sunday, users tend to take less steps compared to other days as it is weekly holiday.

```
dateSummary <- dailyActivity %>%
  group_by(ActivityDate) %>%
  summarize(AvgTotalSteps = mean(TotalSteps))

ggplot(dateSummary, aes(x = ActivityDate, y = AvgTotalSteps, color = "red")) +
  geom_line() +
  geom_smooth()+
  labs(title = "Average Total Steps Over Time",
       x = "Date",
       y = "Average Total Steps")+
  theme_minimal()+
  theme(legend.position = "none")
```

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



The average steps taken over time is seemingly dropping after 1st week of May. Need more investigation here.

Correlation between Total Steps and Weight

```
dailyActivity_steps_Id <- dailyActivity %>% select(Id, TotalSteps)
weightLogInfo_IdW <- read_csv("weightLogInfo.csv") %>% select(Id, WeightKg)
```

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

```
activity_weight <- merge(dailyActivity_steps_Id, weightLogInfo_IdW, by = "Id")

head(activity_weight)
```

```
##           Id TotalSteps WeightKg
## 1 1503960366      12207     52.6
## 2 1503960366      12207     52.6
## 3 1503960366      13019     52.6
## 4 1503960366      13019     52.6
## 5 1503960366      12764     52.6
## 6 1503960366      12764     52.6
```

```
correlation <- cor(activity_weight$TotalSteps, activity_weight$WeightKg)

cat("Correlation between Total Steps and Weight:", correlation, "\n")
```

```
## Correlation between Total Steps and Weight: 0.2647917
```

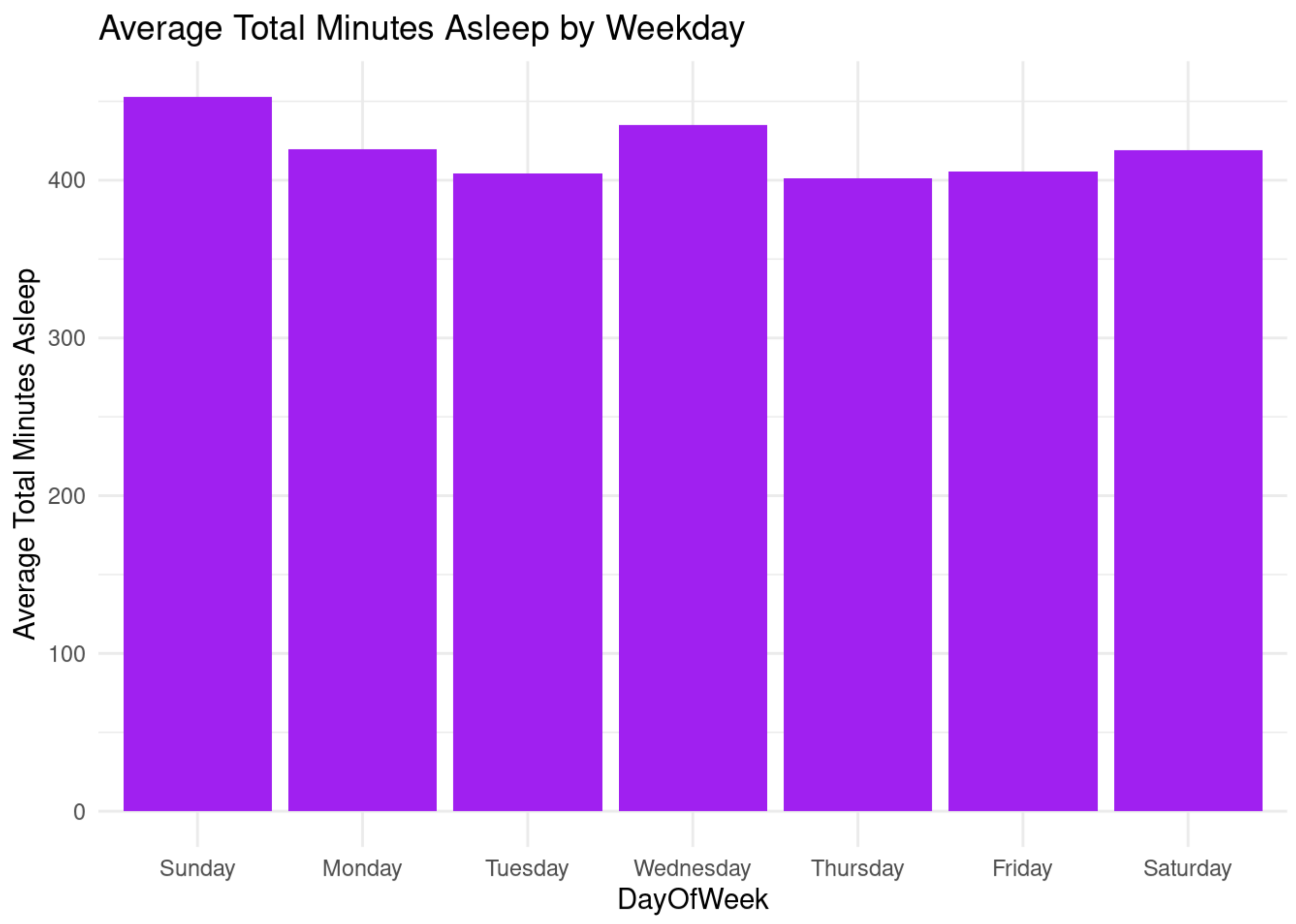
No correlation between total steps walked and the weight of user.

Average Sleep by Weekdays

```
dailyActivitySleep <- read_csv("dailyActivitySleep.csv")
```

```
## Rows: 410 Columns: 18
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl  (17): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesD...
## date  (1): ActivityDate
##
## 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.
```

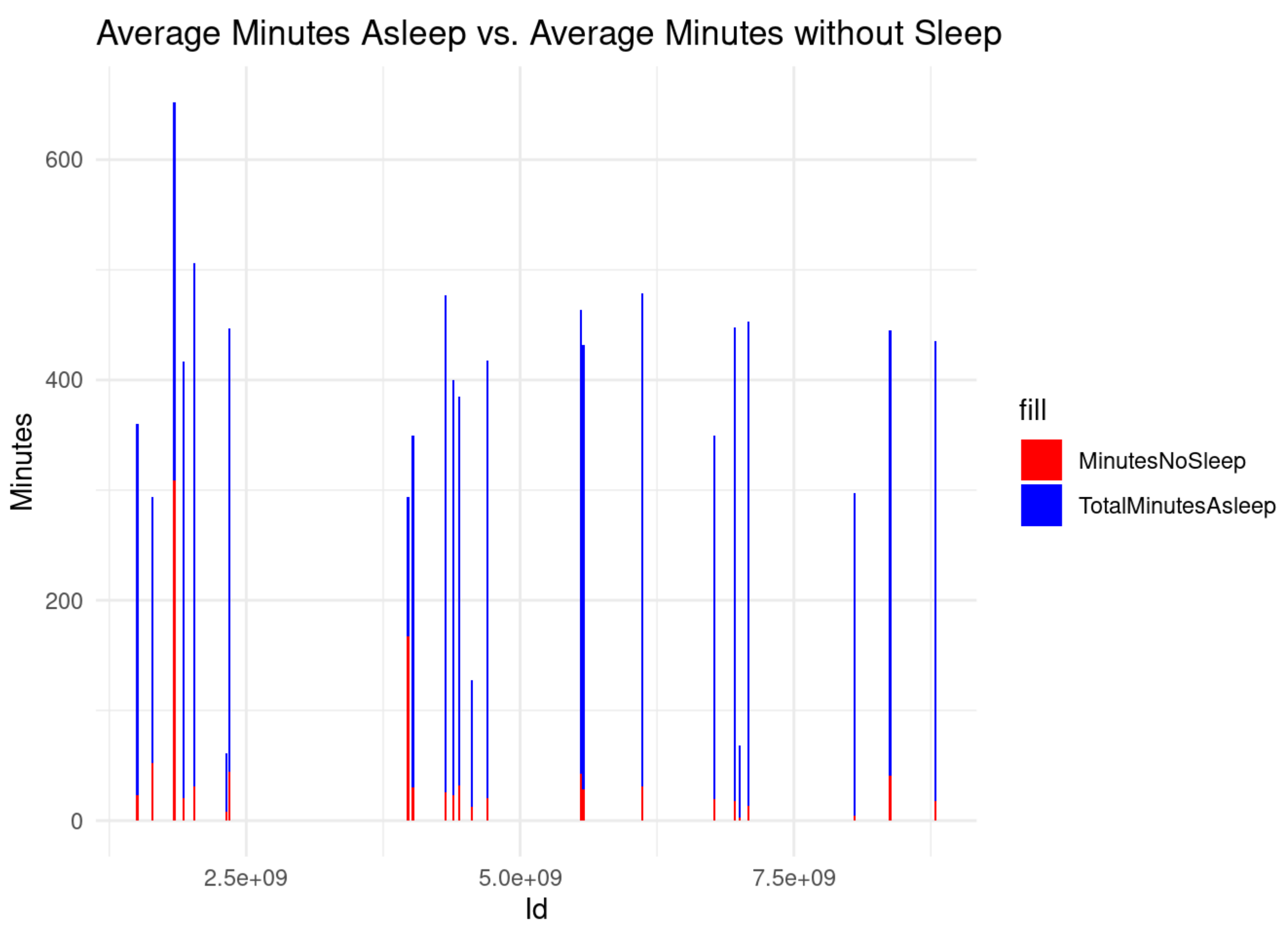
```
dailyActivitySleep$DayOfWeek <- weekdays(dailyActivitySleep$ActivityDate)

dailyAvgSleep <- dailyActivitySleep %>%
  group_by(DayOfWeek) %>%
  summarize(AvgTotalMinutesAsleep = mean(TotalMinutesAsleep))

dailyAvgSleep$DayOfWeek <- factor(dailyAvgSleep$DayOfWeek,
                                      levels = c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturda
y"))

ggplot(dailyAvgSleep, aes(x = DayOfWeek, y = AvgTotalMinutesAsleep)) +
  geom_bar(stat = "identity", fill = "purple") +
  labs(title = "Average Total Minutes Asleep by Weekday", y = "Average Total Minutes Asleep") + theme_minimal()
```



On Sunday, users tend to sleep more compared to other days as it is weekly holiday.

Sleep Time Stats

```
dailyActivitySleep <- dailyActivitySleep %>%
  mutate(MinutesNoSleep = TotalTimeInBed - TotalMinutesAsleep)

average_sleep_data <- dailyActivitySleep %>%
  group_by(Id) %>%
  summarize(AvgMinutesNoSleep = mean(MinutesNoSleep),
            AvgTotalMinutesAsleep = mean(TotalMinutesAsleep))

ggplot(average_sleep_data, aes(x = Id)) +
  geom_bar(aes(y = AvgTotalMinutesAsleep, fill = "TotalMinutesAsleep"), stat = "Identity") +
  geom_bar(aes(y = AvgMinutesNoSleep, fill = "MinutesNoSleep"), stat = "Identity") +
  scale_fill_manual(values = c("TotalMinutesAsleep" = "blue", "MinutesNoSleep" = "red")) +
  labs(title = "Average Minutes Asleep vs. Average Minutes without Sleep",
       x = "Id",
       y = "Minutes") +
  theme_minimal()
```

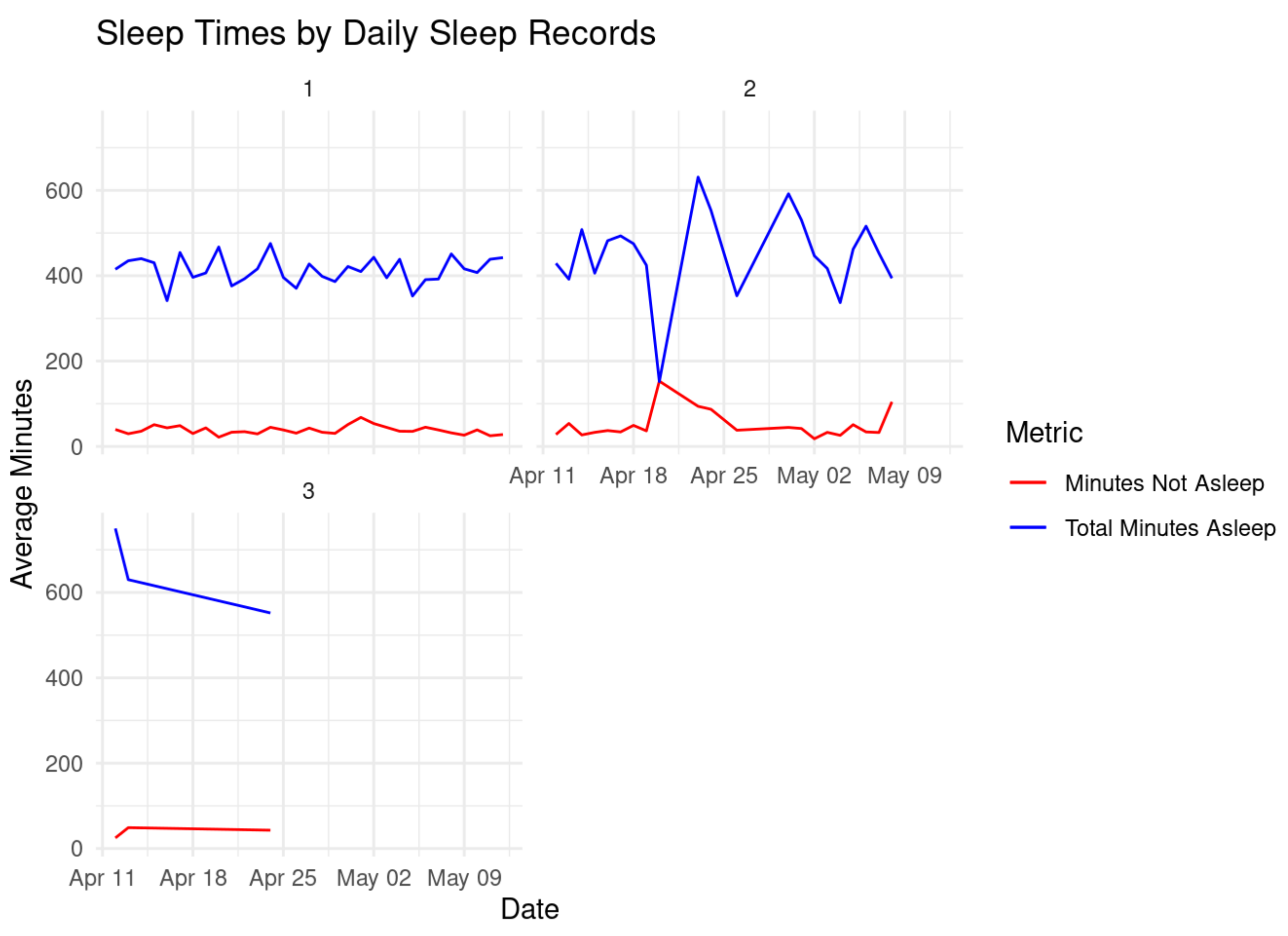


For a some users, the Minutes in bed without sleep is very high compared to minutes asleep. Also, for some, the total time in bed is high.

```
average_sleep_data <- dailyActivitySleep %>%
  group_by(ActivityDate, TotalSleepRecords) %>%
  summarise(
    AvgMinutesNotAsleep = mean(TotalTimeInBed - TotalMinutesAsleep),
    AvgTotalMinutesAsleep = mean(TotalMinutesAsleep)
  )
```

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

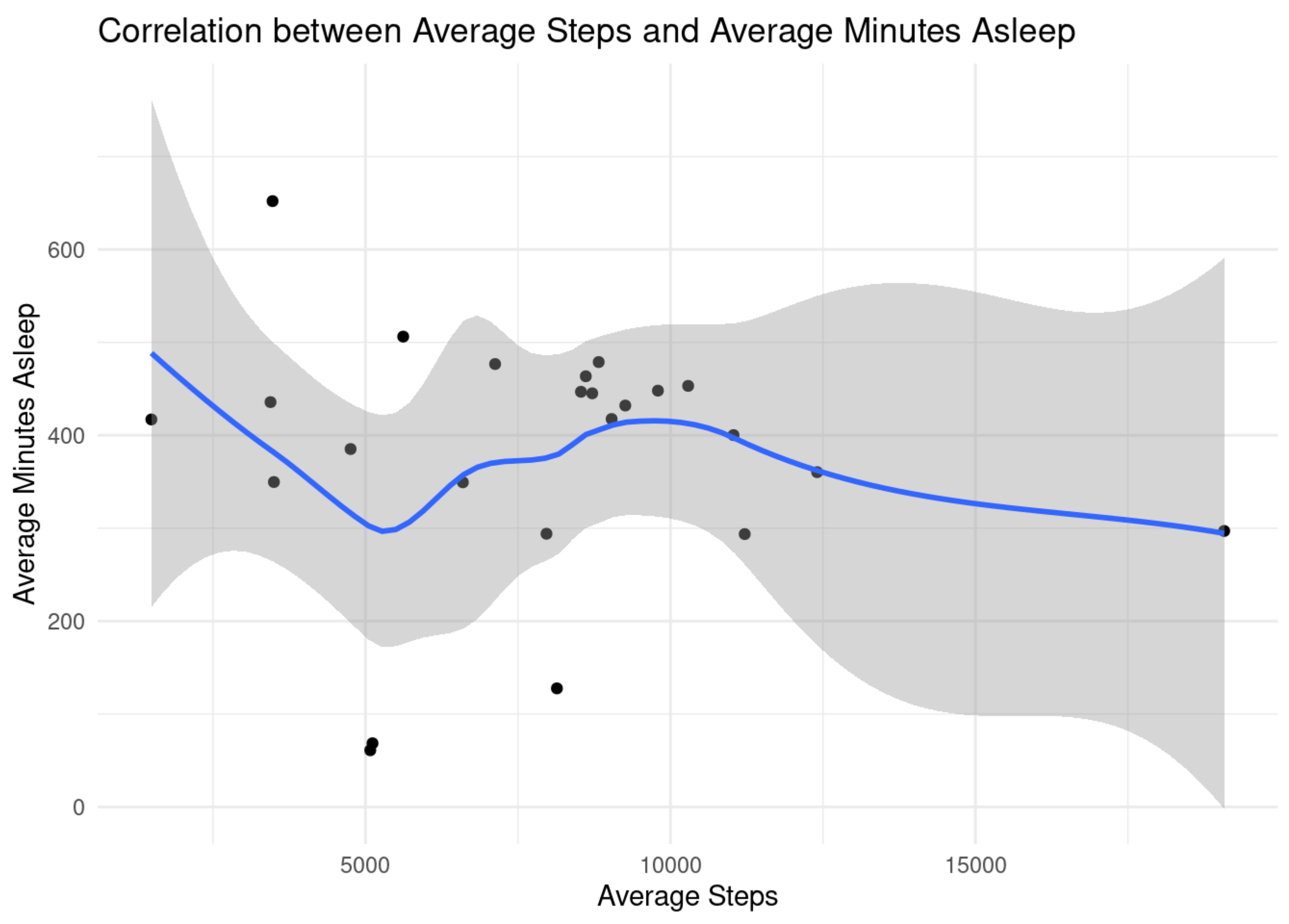
```
sleep_chart <- ggplot(average_sleep_data, aes(x = ActivityDate)) +
  geom_line(aes(y = AvgMinutesNotAsleep, color = "Minutes Not Asleep")) +
  geom_line(aes(y = AvgTotalMinutesAsleep, color = "Total Minutes Asleep")) +
  facet_wrap(~TotalSleepRecords, ncol = 2) +
  labs(title= "Sleep Times by Daily Sleep Records",
    x = "Date",
    y = "Average Minutes",
    color = "Metric"
  ) +
  scale_color_manual(values = c("red", "blue")) +
  theme_minimal()

print(sleep_chart)
```



Average Steps and Average Minutes Asleep

```
# Calculate average steps and minutes asleep for each user
average_steps_sleep <- dailyActivitySleep %>%
  group_by(Id) %>%
  summarize(AvgSteps = mean(TotalSteps), AvgMinutesAsleep = mean(TotalMinutesAsleep))


# Calculate the correlation coefficient
correlation_coefficient <- cor(average_steps_sleep$AvgSteps, average_steps_sleep$AvgMinutesAsleep)

# Create a scatter plot
ggplot(average_steps_sleep, aes(x = AvgSteps, y = AvgMinutesAsleep)) +
  geom_point() +
  geom_smooth() +  # Add a linear regression line
  labs(x = "Average Steps", y = "Average Minutes Asleep") +
  ggtitle("Correlation between Average Steps and Average Minutes Asleep") +
  theme_minimal()
```

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

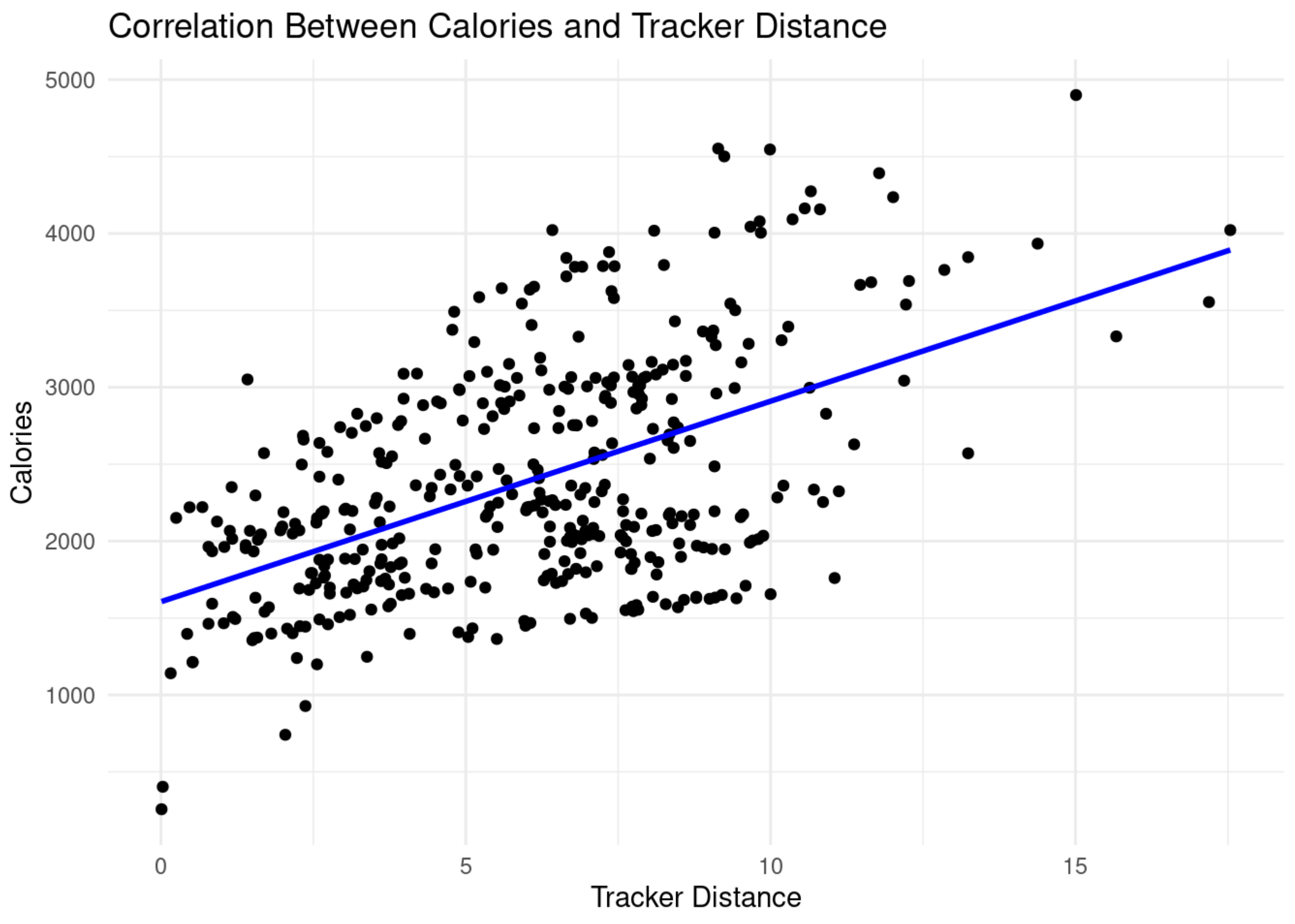


No clear correlation is seen.

Correlation Between Calories and Tracker Distance

```
ggplot(dailyActivitySleep, aes(x = TrackerDistance, y = Calories)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "blue") +
  labs(title = "Correlation Between Calories and Tracker Distance",
       x = "Tracker Distance",
       y = "Calories")+
  theme_minimal()
```

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



Calories burned has a good correlation with the distance recorded in the tracker.

Recommendations

- Promote the importance of maintaining an active lifestyle for weight management along with the products as tools for track of health.
- Run promotions and challenges to motivate users to increase their activity levels and offer rewards for achieving specific step count milestones.
- Bellabeat products can be promoted to Sedentary users with content about the health benefits of increasing their activity.
- Send personalized health tips on weekends along with regular in-general tips.
- Collaborate with sleep experts and provide personalized advice to users with sleep troubles.
- Implement a recommendation engine that suggests activities and sleep tips based on individual user profiles.
- Create infographics and interactive dashboards to make the findings more accessible to users. This can also be a marketing tool to showcase of the products.
- User engagement and feedback should be continuously monitored to adjust marketing strategies and product features so that these align with user needs.