Google Data Analytics Capstone Case Study 2: How can a wellness technology company play it smart?

Adrianna

2024-02-21

About

This document is an a case study for the Google Data Analytics Professional Certificate Program. The case study focuses on the Bellabeat data analysis case study. The case study will follow the following six steps of the data analysis process:

1. Ask
2. Prepare
3. Process
4. Analyze
5. Share
6. Act

Scenario

You are a junior data analyst working on the marketing analyst team at Bellabeat, a high-tech manufacturer of health-focused products for women. Bellabeat is a successful small company, but they have the potential to become a larger player in the global smart device market. Urška Sršen, cofounder and Chief Creative Officer of Bellabeat, believes that analyzing smart device fitness data could help unlock new growth opportunities for the company. You have been asked to focus on one of Bellabeat's products and analyze smart device data to gain insight into how consumers are using their smart devices. The insights you discover will then help guide marketing strategy for the company. You will present your analysis to the Bellabeat executive team along with your high-level recommendations for Bellabeat's marketing strategy.

What is Bellabeat?

Bellabeat is a high-tech company that manufactures health-focused smart products.The company develops 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. Since it was founded in 2013, Bellabeat has grown rapidly and quickly positioned itself as a tech-driven wellness company for women.

Goal

Develop a marketing strategy for Bellabeat based on usage trends and insights discovered in smart device usage and data.

The Task

The team must understand how consumers are using non-Bellabeat smart devices first. Then, the team will select one Bellabeat product to apply those trends and insights uncovered in the non-Bellabeat analysis to help influence a marketing strategy.

The Ask

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?

Deliverable - a report containing the following information:

1. A clear summary of the business task
2. A description of all data sources used
3. Documentation of any cleaning or manipulation of data
4. A summary of your analysis
5. Supporting visualizations and key findings
6. Your top high-level content recommendations based on your analysis

* *The above will be the deliverables of each of the six steps in the data analysis process.*

Ask

Guide for Ask phase

* Problem to be solved:

Understand usage trends and insights from consumers using health-focused smart devices, and use that intel to design a marketing strategy that will effectively market Bellabeat smart device products.

* How insights can drive business decisions:

By understanding usage trends in health-focused smart devices, we can take those insights and build an effective marketing strategy that will entice prospective customers to make Bellabeat's smart devices more appealing, and therefore be more inclined to buy the company's products. An effective market strategy that encourages consumers to buy Bellabeat's products will result in growth opportunities for the company.

Key Tasks

1. Identify the business task:

Understand consumer preferences and usage trends with their health-focused smart devices so Bellabeat can create a marketing strategy for their products that resonates with consumer preferences.

2. Consider key stakeholders:

- **Bellabeat**: A tech company that manufactures health-focused smart products.The company develops technology that informs and inspires women around the world.
- **Urška Sršen**: Bellabeat's cofounder and Chief Creative Officer. Urška is asking the Marketing Analytics team to analyze smart device usage data to gain insights on how Bellabeat can better market its products.
- **Bellabeat Marketing Analytics Team**: A team of data analysts responsible for collecting, analyzing, and reporting data that helps guide Bellabeat's marketing strategy. You joined this team six months ago and have been busy learning about Bellabeat' 's mission and business goals — as well as how you, as a junior data analyst, can help Bellabeat achieve them.

Deliverable

A clear statement of the business task:

*Leverage consumer smart device fitness data to gain insights into how consumers use their devices to unlock growth opportunities for the company.

Prepare

Guide for Prepare phase

* Where is the data located?

The data is provided on an AWS server here. Each data set can be found in a zip file.

* How is the data organized?

There are 18 data sets that record multiple health-related factors, such as activity, calories, workout intensity, sleep, steps, heart rate, and weight.The data sets are go into further granular detail in that each of the mentioned factors are recorded in days, hours, and minutes.The data collected for all data sets is a months worth of data, from 4/12/2026 - 5/12/2016.The data sets are in both long and wide formats.

We will be working with the following data sets:

- Daily Activity
- Daily Calories
- Daily Intensities
- Daily Steps
- Heart Rate Seconds
- Hourly Calories
- Hourly Intensities
- Hourly Steps
- Minute Calories (narrow)
- Minute Calories (wide)
- Minute Intensities (narrow)
- Minute Intensities (wide)
- Minute METs (narrow)
- Minute Sleep

- Minute Steps (narrow)
- Minute Steps (wide)
- Sleep Day
- Weight Log Info

*** Potential issues with bias or credibility in this data: Is the data ROCCC (Reliable, Original, Comprehensive, Current Cited)?**

The data sets were pulled from Kaggle, and it is disclosed that the data points collected are from survey respondents from Amazon Mechanical Turk, a crowdsourcing website where businesses can remotely hire "crowdworkers" to perform on-demand tasks. This means the data is self-reported, which makes it original; however, there are questions about respondent incentives given that they reported the data through Amazon Mechanical Turk, where they would have been hired to perform the work.

***How are you addressing licensing, privacy, security, and accessibility?**

The data sets are publicly available on Kaggle through Mobius and licensed under CC0 Public Domain.

***How was the data's integrity verified?**

- After importing the csv files
- Checked the data types to make sure they made sense and also looked for any null values in each data set

***How does it help you answer your question?**

By having consistent data sets that contain usable data that are free of errors, we can compare the data across all of the data sets and analyze it over the given period of time to identify trends among consumers' smart device usage, including (but not limited to) consumers' sleeep, activity duration, level of intensity, steps taken, heart rates, and calories burned by minute, hour day, etc.

*Are there any problems with the data?

Depending on if the users leveraged their smart devices correctly to collect the data, it would be fair to assume that the data is as accurate as the devices allow (i.e. the total steps taken in a day should be accurate if the survey respondent used their device appropriately). It also appears that users need to be diligent in recording their information every day for the given time period so the data is complete per user. However, there were only 30 respondents providing their data, which may be too small of a sample size to ascertain meaningful results. Moreover, it is stated that the data set collates data used from various types of fitness trackers from users, so users may have recorded their output with different devices, so there are questions about consistency of the data collected.

Additionally, the data doesn't offer enough information about each respondent's profile, which would ultimately play a role in how the data should be interpreted and recommendations that Bellabeat can offer. Some factors include gender and age, where the level of intensity, steps taken, and caloric intake provided in the data sets can yield different results depending on these demographic factors, especially given that Bellabeat's target audience is women, so it would be helpful to definitively confirm if the data collected from the survey respondents for other health smart devices are men or women.

Lastly, it is also important to consider where respondents reside, as cultural and lifestyle can be influenced by one's geolocation. This information is not availale so there is potential that the data may be skewed.

Key tasks

1. Download data and store it appropriately

Since this is a personal project, I downloaded the zip files for each month's trip data and unzipped them in a separate folder in my OneDrive. I then created a subfolder to store all the csvs.

2. Identify how it's organized

The data is broken out into multiple data sets by different health factors recorded, such as heart rate, steps, workout intensity, etc. Those health factors are further broken out by minutes, hours, and days for each survey respondent.

3. Sort and filter the data

I will use various functions to arrange and filter the data sets if necessary as outlined in this document.

4. Determine the credibility of the data

The data sets were pulled from Kaggle, and it is disclosed that the data points collected are from survey respondents from Amazon Mechanical Turk, a crowdsourcing website where businesses can remotely hire "crowdworkers" to perform on-demand tasks. This means the data is self-reported, which makes it original; however, there are questions about respondent incentives given that they reported the data through Amazon Mechanical Turk, where they would have been hired to perform the work. The data is considered third party data, as it was provided from outside sources (Kaggle) who didn't collect it directly.

Deliverable

A description of all data sources used:

I used all 18 data sets, so 18 csv data sets were initially pulled into R for analysis. The files contain data from the the period between 4/12/2016 through 5/12/2016. Licensing, privacy, and accessibility matters for the data source were addressed in the guide above.

See code used to prepare the data below:

Data sets included in this exercise are as follows:

dailyactivity
dailycalories
dailyintensities
dailysteps
heartrate_seconds
hourlycalories
hourlyintensities
hourlysteps
minutecaloriesnarrow
minutecalorieswide
minuteintensitiesnarrow
minuteintensitieswide
minutemetsnarrow

minutesleep
minutestepsnarrow
minutestepswide
sleepday
weightloginfo

```
# Import all csv files to create your data sets and variables and view each one.

dailyactivity <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert - Ca
View(dailyactivity)

dailycalories <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert - Ca
View(dailycalories)

dailyintensities <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert -
View(dailyintensities)

dailysteps <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert - Capst
View(dailysteps)

heartrate_seconds <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert
View(heartrate_seconds)

hourlycalories <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert - C
View(hourlycalories)

hourlyintensities <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert
View(hourlyintensities)

hourlysteps <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert - Caps
View(hourlysteps)

minutecaloriesnarrow <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Ce
View(minutecaloriesnarrow)

minutecalorieswide <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert
View(minutecalorieswide)

minuteintensitiesnarrow <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics
View(minuteintensitiesnarrow)

minuteintensitieswide <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics C
View(minuteintensitieswide)

minutemetsnarrow <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert -
View(minutemetsnarrow)

minutesleep <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert - Caps
View(minutesleep)

minutestepsnarrow <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert
```

```
View(minutestepsnarrow)

minutestepswide <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert -
View(minutestepswide)

sleepday <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert - Capsto
View(sleepday)

weightloginfo <- read.csv('C:/Users/Adj/OneDrive/Documents/Data Projects/Google Data Analytics Cert - C
View(weightloginfo)
```

These will be our original variables to start with (x18) * It should be noted that all time-based data for all data sets have *chr* data types. We will need to transform them to *date* or *date time* data types so we can work with the data. Transformation will occur in the *Process* phase of the data analysis cycle.

Process

Guide for Process phase

* What tools were used in this case study and why?

I have chosen to use R for this exercise, as the sheer volume of records in each data set may not be compatible with spreadsheets or other data analysis tools, such as Microsoft Excel or an SQL database. Furthermore, transforming and visualizing the data will be integral in getting insights from the data, and R's built-in functions, library packages, and visualization features will allow me to clean and transform the data as necessary, and create sufficient charts/graphs. I will be able to do both the data analysis and data visualization within the same application.

* How the data's integrity was verified:

- Transformed the data by:
- Checked each data set to see if the same unique ids appeared across all data sets (i.e. if less than exepcted unique values, duplibates may be present)
- Checked for duplicates in the data sets and deleted any rows with duplicate ride IDs
- Removed any rows where the data didn't make sense
- Cleaned up column names so the aliases are formatted consistently and are labeled intuitively
- Checked for data type consistency in each data set and converted columns data type if needed (i.e.Convert date columns so they are consistent across time stamps. Example would be "daily" data sets are consistent, "hourly" data sets are consistent, etc.)
- Transformed data so that they make sense in context and will enable me to work with the data (i.e. dates are of the correct data type)

* Steps taken to ensure that the data is clean:

The remainder of this document will outline the methods I applied to ensure data integrity among the data sets. This includes (but is not limited to) checking for duplicates, data entry errors, null values, consistency across data sets, whether the data points are valid and makes sense, etc.

*** Verify the data so that it is clean and ready to analyze:**

The result of each step in the data-cleaning process will be outlined in the documentation after each function is executed.

*** Document the cleaning process so results can be reviewed and shared:**

This file will document the cleaning process and all data transformation performed to get the data sets to a point where they are usable to gain insights from.

Key Tasks

1. Check the data for errors

2. Choose tools

3. Transform the data so it can be used effectively

4. Document the cleaning process

Deliverable

Documentation of any cleaning or manipulation of data

The code chunks below document all functions used to clean and manipulate the data sets to check for errors and transform the data so that it be used effectively for further analysis. Comments in the code chunks highlight the intent of the functions used to clean and/or manipulate the data. Findings are summarized below:

```
# Check that each data set has the same number of unique IDs.

n_unique(dailyactivity$id)
n_unique(dailycalories$Id)
n_unique(dailyintensities$Id)
n_unique(dailysteps$Id)
n_unique(heartrate_seconds$Id)
n_unique(hourlycalories$Id)
n_unique(hourlyintensities$Id)
n_unique(hourlysteps$Id)
n_unique(minutecaloriesnarrow$Id)
n_unique(minuteintensitieswide$Id)
n_unique(minutemetsnarrow$Id)
n_unique(minuteintensitieswide$Id)
n_unique(minutemetsnarrow$Id)
n_unique(minutesleep$Id)
n_unique(minutestepsnarrow$Id)
n_unique(minutestepswide$Id)
n_unique(sleepday$Id)
n_unique(weightloginfo$Id)
```

Most data sets returned 33 unique user IDs. Those that returned otherwise are the “sleepday” and “weightloginfo” data sets, indicating that not all respondents provided all of their sleep activity or weight info. The

"weightloginfo" data set in particular to too small of a sample size with only 8 users to gain meaningful insights from.

```
# Check for duplicate records in each data set

sum(duplicated(dailyactivity))
sum(duplicated(dailycalories))
sum(duplicated(dailyintensities))
sum(duplicated(dailysteps))
sum(duplicated(heartrate_seconds))
sum(duplicated(hourlycalories))
sum(duplicated(hourlyintensities))
sum(duplicated(hourlysteps))
sum(duplicated(minutecaloriesnarrow))
sum(duplicated(minutecalorieswide))
sum(duplicated(minuteintensitiesnarrow))
sum(duplicated(minuteintensitieswide))
sum(duplicated(minutemetsnarrow))
sum(duplicated(minutesleep))
sum(duplicated(minutestepsnarrow))
sum(duplicated(minutestepswide))
sum(duplicated(sleepday))
sum(duplicated(weightloginfo))
```

* *If return value > 0, then there is a duplicate ID in the data set*

The "minutesleep" and "sleepday" data sets returned values greater than 0, indicating 543 instances of identical rows and 3 instances of identical rows for "minutesleep" and "sleepday" data sets, respectively.

```
# Return duplicated records from data sets that returned duplicate IDs (optional)

get_dupes(minutesleep)
```

```
## No variable names specified - using all columns.
```

```
get_dupes(sleepday)
```

```
## No variable names specified - using all columns.
```

This returns a view of the duplicated rows in each data set. The biggest tell that records are duplicates are that the are duplicate time-related columns for the same exact user id. The total minutes slept and total time in bed are also identical, further confirming that these are duplicated entries and need to be scrubbed so that only one record of each instance remains in the data set.

```
# Remove duplicate rows from the two data sets that returned dupes - create "cleaned", new variable nam

minutes_sleep <- minutesleep %>%
  distinct() %>%
  drop_na()

sleep_day <- sleepday %>%
  distinct() %>%
  drop_na()
```

```r
# Test new variables and check that duplicate function returns "0" to confirm that duplicate row values

sum(duplicated(minutes_sleep))

sum(duplicated(sleep_day))
```

Duplicate records in each data set were successfully removed. The data sets should be clean to work with and perform further analysis.

```r
# Clean up column names (column names to lower case) and revise variable names (add underscores between

clean_names(dailyactivity)
daily_activity <- rename_with(dailyactivity, tolower)

View((daily_activity))
```

```r
clean_names(dailycalories)
daily_calories <- rename_with(dailycalories,tolower)

View(daily_calories)
```

```r
clean_names(dailyintensities)
daily_intensities<- rename_with(dailyintensities, tolower)

View(daily_intensities)
```

```r
clean_names(dailysteps)
daily_steps <- rename_with(dailysteps, tolower)

View(daily_steps)
```

```r
clean_names(heartrate_seconds)
heart_rate_seconds <- rename_with(heartrate_seconds, tolower)

View(heart_rate_seconds)
```

```r
clean_names(hourlycalories)
hourly_calories <- rename_with(hourlycalories, tolower)

View(hourly_calories)
```

```r
clean_names(hourlyintensities)
hourly_intensities <- rename_with(hourlyintensities, tolower)

View(hourly_intensities)
```

```r
clean_names(hourlysteps)
hourly_steps <- rename_with(hourlysteps, tolower)

View(hourly_steps)
```

```
clean_names(minutecaloriesnarrow)
minute_calories_narrow <- rename_with(minutecaloriesnarrow, tolower)

View(minute_calories_narrow)
```

```
clean_names(minutecalorieswide)
minute_calories_wide <- rename_with(minutecalorieswide, tolower)
minute_calories_wide <- minute_calories_wide %>%
  rename(activityminute = activityhour)

View(minute_calories_wide)
```

```
clean_names(minuteintensitiesnarrow)
minute_intensities_narrow <- rename_with(minuteintensitiesnarrow, tolower)

View(minute_intensities_narrow)
```

```
clean_names(minuteintensitieswide)
minute_intensities_wide <- rename_with(minuteintensitieswide, tolower)
minute_intensities_wide <- minute_intensities_wide %>%
  rename(activityminute = activityhour)

View(minute_intensities_wide)
```

```
clean_names(minutemetsnarrow)
minute_mets_narrow <- rename_with(minutemetsnarrow, tolower)

View(minute_mets_narrow)
```

```
clean_names(minutes_sleep)
minute_sleep <- rename_with(minutes_sleep, tolower)

View(minute_sleep)
```

```
clean_names(minutestepsnarrow)
minute_steps_narrow <- rename_with(minutestepsnarrow, tolower)

View(minute_steps_narrow)
```

```
clean_names(minutestepswide)
minute_steps_wide <- rename_with(minutestepswide, tolower)
minute_steps_wide <- minute_steps_wide %>%
  rename(activityminute = activityhour)

View(minute_steps_wide)
```

```
clean_names(sleep_day)
sleep_day <- rename_with(sleep_day,tolower)

View(sleep_day)
```

```r
clean_names(weightloginfo)
weight_log_info <- rename_with(weightloginfo, tolower)

View(weight_log_info)
```

All data sets are renamed as a second version by adding underscores after each word in the data set name for consistency. Column names in each data set are now in lower case form for consistency as well.

```r
# Convert date columns so they are consistent across time stamps (i.e. "daily" data sets are consistent

# Check the structure of the data sets to ensure the date columns are the correct data type.

daily_activity <- daily_activity %>%
  rename(date = activitydate) %>%
  mutate(date = as_date(date, format = "%m/%d/%Y"))

str(daily_activity)
```

```r
sleep_day <- sleep_day %>%
  rename(date = sleepday) %>%
  mutate(date = as_date(date, format = "%m/%d/%Y %I:%M:%S %p", tz = Sys.timezone()))
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `date = as_date(date, format = "%m/%d/%Y %I:%M:%S %p", tz =
##   Sys.timezone())`.
## Caused by warning:
## ! `tz` argument is ignored by `as_date()`
```

```r
str(sleep_day)
```

```r
hourly_calories <- hourly_calories %>%
  rename(date_time = activityhour) %>%
  mutate(date_time = as.POSIXct(date_time, format = "%m/%d/%Y %I:%M:%S %p", tz = Sys.timezone()))

str(hourly_calories)
```

```r
hourly_intensities <- hourly_intensities %>%
  rename(date_time = activityhour) %>%
  mutate(date_time = as.POSIXct(date_time, format = "%m/%d/%Y %I:%M:%S %p", tz = Sys.timezone()))

str(hourly_intensities)
```

```r
hourly_steps <- hourly_steps %>%
  rename(date_time = activityhour) %>%
  mutate(date_time = as.POSIXct(date_time, format = "%m/%d/%Y %I:%M:%S %p", tz = Sys.timezone()))

str(hourly_steps)
```

Date columns in data sets have been transformed as appropriate to either Date or Datetime data types so they are consistent with the rest of the data sets. The transformed dates in the above columns were

specifically selected because their "date" data types needed to match the formats for the rest of the data sets by minute, hourly, or daily data sets for consistency. Column names related to dates were renamed for clarity.

```
# Join data sets by "daily", "hourly", "minutes" using *merge* to consolidate the number of data sets:

daily_sleep_activity_merged <- merge(daily_activity, sleep_day, by = c("id", "date"))

View(daily_sleep_activity_merged)
```

* *You may need to merge the daily, hourly, or minutes data sets multiple times*

```
hourly_step_intensities_calories <- merge( x= hourly_steps,
                                           y = hourly_intensities,
                                           by.x = c("id", "date_time"),
                                           by.y = c("id", "date_time"),
                                           all = TRUE)

View(hourly_step_intensities_calories)
```

```
# Merge third data set to *hourly_step_intensities_calories*

hourly_step_intensities_calories <- merge(x = hourly_step_intensities_calories,
                                          y = hourly_calories,
                                          by.x = c("id", "date_time"),
                                          by.y = c("id", "date_time"),
                                          all = TRUE)

View(hourly_step_intensities_calories)
```

```
minutes_steps_mets_intensities_calories_narrow <- merge(x = minute_calories_narrow,
                                                        y = minute_intensities_narrow,
                                                        by.x = c("id", "activityminute"),
                                                        by.y = c("id", "activityminute"),
                                                        all = TRUE)

View(minutes_steps_mets_intensities_calories_narrow)
```

```
# Merge third data set to *minutes_steps_mets_intensities_calories_narrow*

minutes_steps_mets_intensities_calories_narrow <- merge(x = minutes_steps_mets_intensities_calories_nar
                                                        y = minute_mets_narrow,
                                                        by.x = c("id", "activityminute"),
                                                        by.y = c("id", "activityminute"),
                                                        all = TRUE
                                                        )

View(minutes_steps_mets_intensities_calories_narrow)
```

```
# Merge fourth data set to *minutes_steps_mets_intensities_calories_narrow*

minutes_steps_mets_intensities_calories_narrow <- merge(x = minutes_steps_mets_intensities_calories_nar
```

```
                                                    y = minute_steps_narrow,
                                                    by.x = c("id", "activityminute"),
                                                    by.y = c("id", "activityminute"),
                                                    all = TRUE)

View(minutes_steps_mets_intensities_calories_narrow)
```

```
minute_steps_intensities_calories_wide <- merge(x = minute_calories_wide,
                                                y = minute_intensities_wide,
                                                by.x = c("id", "activityminute"),
                                                by.y = c("id", "activityminute"),
                                                all = TRUE)

View(minute_steps_intensities_calories_wide)
```

```
# Merge third data set to *minute_steps_intensities_calories_wide*

minute_steps_intensities_calories_wide <- merge(x = minute_steps_intensities_calories_wide,
                                                y = minute_steps_wide,
                                                by.x = c("id", "activityminute"),
                                                by.y = c("id", "activityminute"),
                                                all = TRUE)

View(minute_steps_intensities_calories_wide)
```

Summarize data sets

Daily data set summaries:

```
daily_activity %>%
  summary()

sleep_day %>%
  summary()

daily_calories %>%
  select(calories) %>%
  summary()

daily_intensities %>%
  summary()

daily_steps %>%
  select(steptotal) %>%
  summary()


  # "hourly" data set summaries
hourly_calories %>%
  select(calories) %>%
  summary()
```

```
hourly_intensities %>%
  select(totalintensity,
         averageintensity) %>%
  summary()

hourly_steps %>%
  select(steptotal) %>%
  summary()
```

Minutes data set summaries:

```
minute_calories_narrow %>%
  select(calories) %>%
  summary()

minute_intensities_narrow %>%
  select(intensity) %>%
  summary()

minute_steps_narrow %>%
  select(steps) %>%
  summary()

minute_mets_narrow %>%
  select(mets) %>%
  summary()

minute_calories_wide %>%
  summary()

minute_intensities_wide %>%
  summary()

minute_steps_wide %>%
  summary()
```

Summarize remaining data sets (heart rate seconds, weight log info):

```
heart_rate_seconds %>%
  select(value) %>%
  summary()

weight_log_info %>%
  select(weightkg,
         weightpounds,
         fat,
         bmi) %>%
  summary()
```

```
# View the structure of the merged data sets and note any inconsistencies in data types, if present:

str(hourly_step_intensities_calories)
```

```r
str(minutes_steps_mets_intensities_calories_narrow)
str(minute_steps_intensities_calories_wide)
str(daily_sleep_activity_merged)
```

The *mintes_steps_mets_intensities_calories_narrow* and *minute_steps_intensities_calories_wide* data sets still have *chr* data types for the time columns, but this is acceptable because it is in a readable format that we need.

Extra summary data sets (group_by):

```r
daily_activity_summary_group <- daily_activity %>%
  group_by(id) %>%
  summarise(mean_calories = mean(calories),
            avg_steps = mean(totalsteps),
            avg_distance = mean(totaldistance))

View(daily_activity_summary_group)
```

Analyze

Guide for Analyze phase

*How should you organize your data to perform analysis on it?

The 18 data sets were validated and cleaned/transformed, then merged as necessary in the *Prepare* and *Process* phases. It made the most sense to group the data sets together that were recorded by minute, hour, day, etc. so that we could have consolidated data sets to work with for all health factors and time stamps. With fewer (but larger) data frames, this made it easier to perform any analysis so we can spot trends and extract insights from a holistic view of the time frames for which the data is captured.

*Has your data been properly formatted?

Yes, the data was formatted and transformed for consistency in the *Process* phase. Most of these steps occurred before the data sets were merged (so we could merge the data sets to begin with).

*What surprises did you discover in the data?

There were inconsistencies in the formatting and data types of some columns across the 18 data sets before they were merged, particularly the date/time columns. There were also duplicate entries for some users' responses in only some of the data sets.

*What trends or relationships did you find in the data?

This will be outlined in the code chunks and subsequent summary below.

*How will these insights help answer your business questions?

We expect the insights gathered from the analysis to provide us a better understanding as to how health-focused smart device users are leveraging their devices. From there, we can create a targeted approach to market Bellabeat's own smart devices to consumers, and therefore expand growth within the company.

Key tasks

1. Aggregate your data so it's useful and accessible

2. Organize and format your data

3. Perform calculations

4. Identify trends and relationships

Deliverable

A summary of your analysis.

The graphs and data transformations/manipulations produced the following insights:

- Steps and calories per day appear to be correlated, where the more steps survey respondents recorded, the more calories they burned. Most users tended to take between 5,000 to 15,000 steps per day.
- Uses slept an average of 6.99 hours throughout the week. The days that users averaged the most sleep was on Sunday and Wednesday, respectively. This is below the recommended 8 hours per night for most people. Perhaps Bellabeat's smart device can notify users if their sleep levels are below optimal.
- Users appeared to get in the most steps on Saturday, then Monday and Tuesday, with an overall average of 8,533 steps. Sunday had the least steps recorded for users at 7,298 steps, which isn't surprising because most people tend to take Sunday as a rest day in their routine. Saturdays, which averaged 9,871 steps, may be the most active because people tend to complete errands on the weekend for tasks that may have built up over the week that couldn't be completed during the work week. Mondays and Tuesdays might be seeing high activity due to the start of the week usually kicking off quickly for many people in most jobs. The number of recommended steps per day is 10,000, so most users are not getting in enough steps for the day. Saturday is the only day where users tend to come close but still fall short. To address this, Bellabeat smart devices can create some sort of incentive to encourage users to hit their daily target.
- The highest caloric intake among users was on Saturday at 2,507. This correlates with the scatterplot findings in that the more steps consumers took throughout the day, the more calories they also consumed, as Saturdays had the highest step count as well. Another factor to consider for the higher calorie consumption on Saturdays is that it is the beginning of the weekend where people tend to let loose after a long week and may treat themselves.
- Distance walked per day is also highest on the same days of the week as most steps taken by day of week. This is intuitive, as steps taken and distance traveled are strongly correlated, as also illustrated by the scatterplot visualizing *Total Steps vs Total Distance*. Assuming the distance data is recorded in miles, the average distance walked throughout the week is approximately 6 miles.
- The average time spent in bed per night is 7.7 hours. Sunday was the highest at 8.3 hours, and Wednesday followed with 7.8 hours. This strongly correlates with daily sleep. Perhaps people are able to climb into bed earlier on a Sunday and want to feel rested and energized to kick off the week.
- Steps and intensity seem correlate more strongly in the beginning but intensity levels appear to taper off after a certain number of steps are taken, showing diminishing returns after around the 3,750 step mark. Beyond that point, the number of steps taken don't appear to be as strongly correlated with intensity, which could possibly be explained by fatigue after so many steps are taken where people cannot keep up the same intensity after a while.

- The peak intensity recorded among users is between 16:00 and 18:00 hours (4pm and 6pm), with the highest intensity throughout the day at 17:00 hours (5pm). This time of the day indicates that users may be upping their activity and working out in the early evening after work. The second highest peak in intensity throughout the day happened around 11:00 and 13:00, which may be attributed to users trying to squeeze in a mid-day workout during typical lunch break time during the work day. Lowest activity occurred in the week hours of the morning within the first four hours after midnight, which aligns with most people sleep time.
- Average steps per hour lines up with average intensity hour as expected. If users are upping their intensities, they will naturally take more steps due to increased activity. Peak and lowest average steps directly correlate to the peak and lowest average intensities throughout the day.
- Half of respondents are classified as "low users", meaning they used their devices less than 10 days during the recorded period. Moderate usage was lowest among respondents, which falls between 11 and 20 days of use. This shows that there is a gap in users leveraging their smart devices consistently, so there is an opportunity for Bellabeat to encourage more use and create a market strategy around that. However, "Daily Activity" needs to be clearly defined. Here, we are assuming respondents did not record any activity or health factors at all.

* *See Appendix for code used to analyze the data below.*

Share

Guide for Share phase

*** Were you able to answer the business questions?**

We were able to extract some insights and trends in the data; however, this data set is still too small to be representative of the general population or target audience. Many of the trends found in the data seem to be in line with typical behavior revolving around physical activity.

*** What story does your data tell?**

The data shows seems to explain physical activity-related behaviors that coincide with your average adult, such as peak times of the day or week when activity occurred. Some the data showed that many respondents in were not getting the recommended activity or sleep, and may be over-consuming calories based on the data reported. This is understandable given the busy lifestyles that most adults typically live, assuming the respondents are U.S.-based.

*** How do your findings relate to your original question?**

This highlights an opportunity for Bellabeat to create features in their smart devices that will address the shortcomings seen in the data, in which an informative marketing campaign can be focused on encouraging more frequent usage of the devices, and incentives to encourage consumers to reach milestones for each health factor to optimize health and well-being.

*** Who is your audience? What is the best way to communicate with them?**

Our main stakeholder is Urška Sršen, Bellabeat's cofounder and CCO, and the executive team who will take the analysis findings and use it to make a marketing strategy for Bellabeat's smart devices come to fruition.

It is also important to consider my own peers in the Marketing Analytics team who are also playing a part in this analysis.

Given that the main stakeholders of my audience are at the executive-level, creating a succinct and straightforward deliverable to convey my findings would be idea, where information is communicated at a high level and granularity in the technical analysis used to produce the analysis are generally subdued or avoided to reduce presenting unnecessary information. The final deliverable would ideally be communicated in slides and dashboards to show a high-level summary of findings and recommendations.

*** Can data visualization help you share your findings?**

Yes, data visualization will be key in presenting my findings, as they will be able to explicitly call out key findings that jump out in the data.

*** Is your presentation accessible to your audience?**

The presentation should only be available to the designated audience. The deliverable formats that we will use to present our findings (Powerpoint slides and Tableau dashboards) can be made accessible to our intended audience, with safety controls to maintain confidentiality.

Key Tasks

1. **Determine the best way to share your findings**
2. **Create effective data visualizations**
3. **Present your findings**
4. **Ensure your work is accessible**

Deliverable

Supporting visualizations and key findings.

* *Find the code for visual aids in the appendix below.*

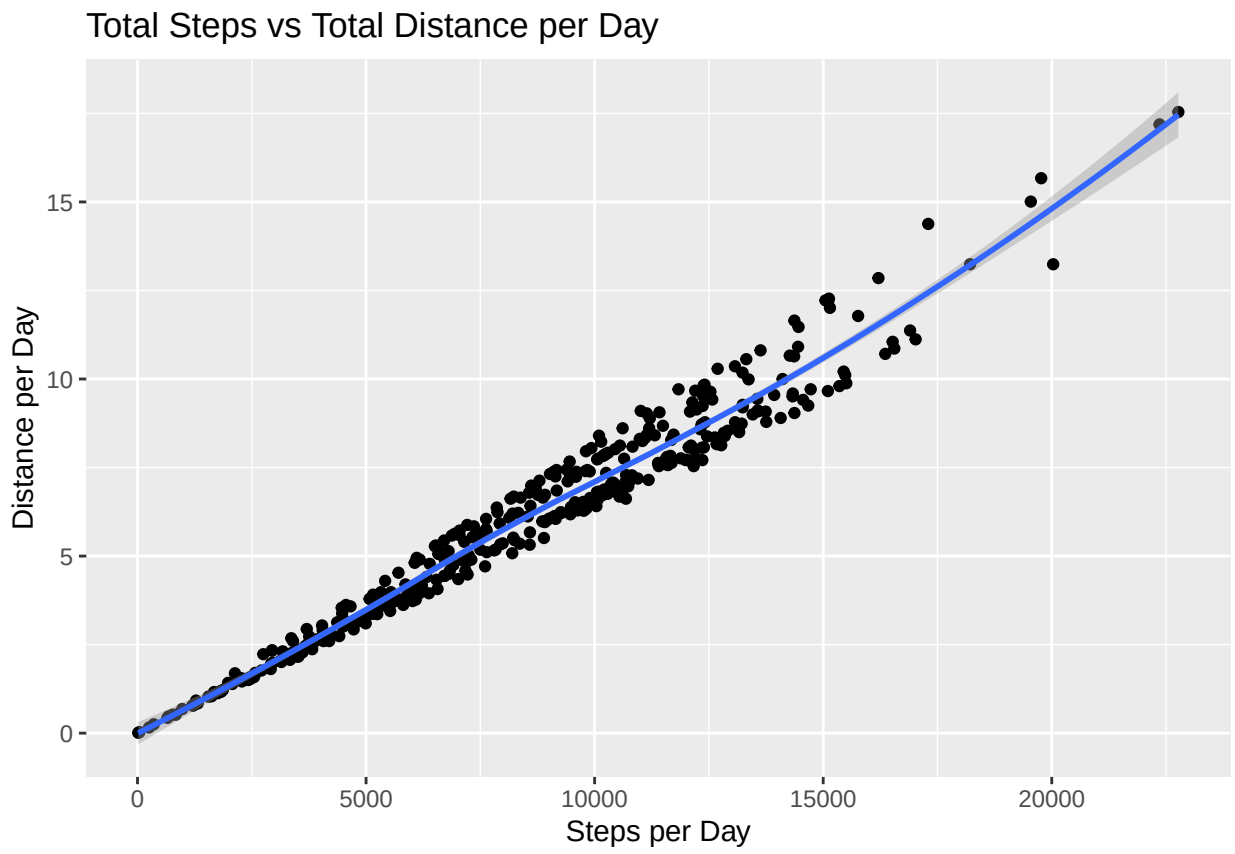
Appendix

```
# Daily steps vs calories - scatterplot

ggplot(data = daily_sleep_activity_merged, aes(x = totalsteps, y = calories)) +
  geom_point() +
  geom_smooth() +
  labs(title = "Total Steps vs Calories per Day",
       x = "Steps per Day",
       y = "Calories per Day")
```

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


```
# Daily distance vs steps - scatterplot

ggplot(data = daily_sleep_activity_merged, aes(x = totalsteps, y = totaldistance)) +
  geom_point() +
  geom_smooth() +
  labs(title = "Total Steps vs Total Distance per Day",
       x = "Steps per Day",
       y = "Distance per Day")
```

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

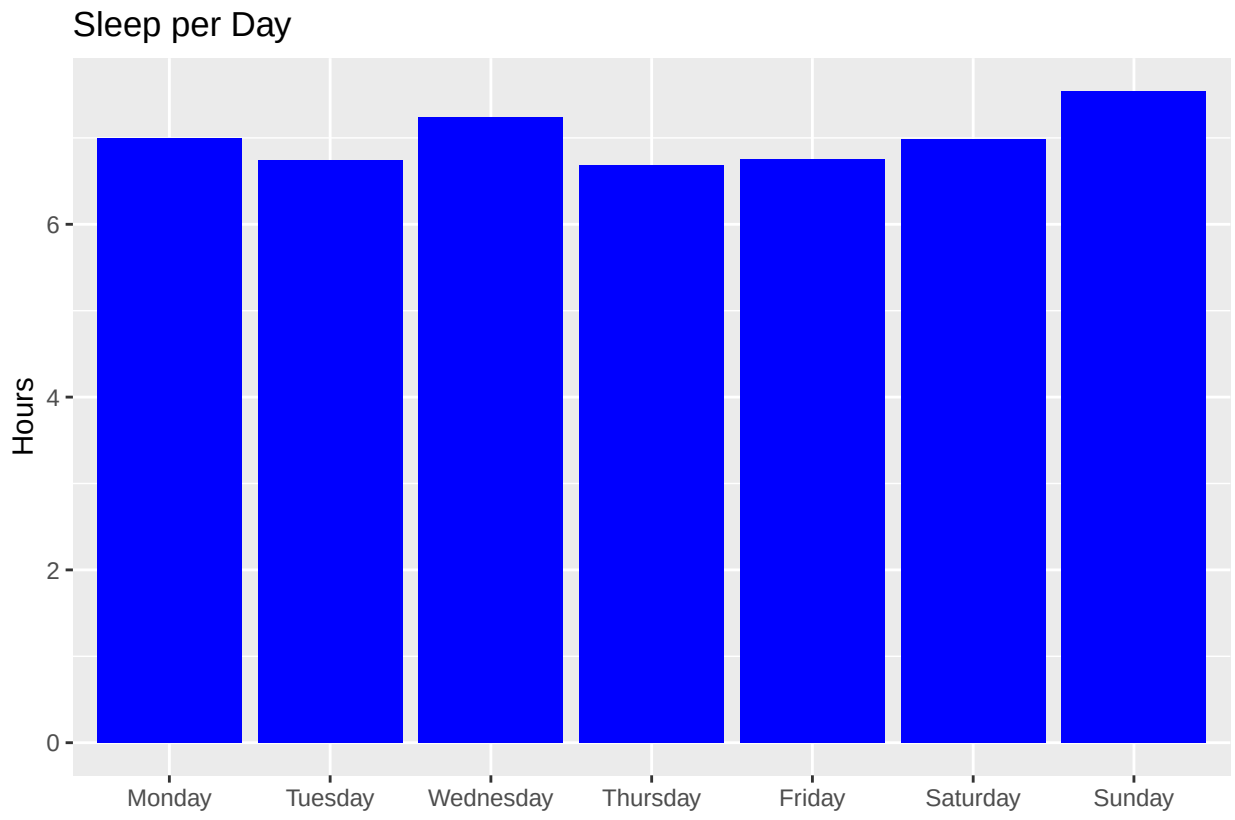



```
# Add 'day of week" names to "daily" data sets as a new column

weekday_sleep <- daily_sleep_activity_merged %>%
  mutate(weekday = weekdays(date))
weekday_sleep$weekday <- ordered(weekday_sleep$weekday, levels=c("Monday", "Tuesday", "Wednesday", "Thu

View(weekday_sleep)
```

```
# Group total minutes of sleep per day by day of week
weekday_sleep_summary_group <- weekday_sleep %>%
  group_by(weekday) %>%
  summarise(daily_sleep = mean(totalminutesasleep),
            daily_steps = mean(totalsteps),
            daily_distance = mean(totaldistance),
            daily_bedtime = mean(totaltimeinbed),
            daily_calories= mean(calories)
            )

View(weekday_sleep_summary_group)
```

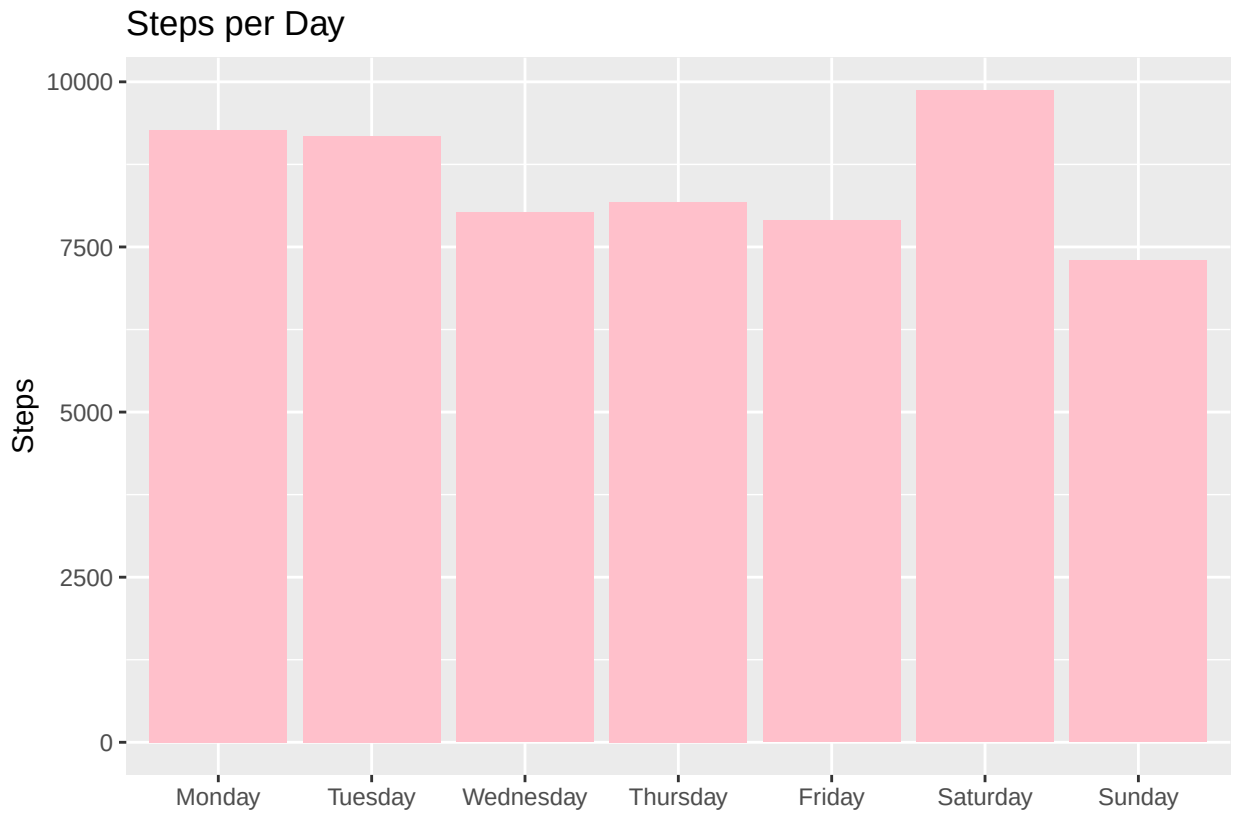
```
# Hours of sleep by day of week

# Graph variables by day of week

ggplot(data = weekday_sleep_summary_group, aes(x = weekday, y = daily_sleep/60)) +
```

```
  geom_col(fill = "blue") +
  labs(title = "Sleep per Day",
       x = "",
       y = "Hours")
```



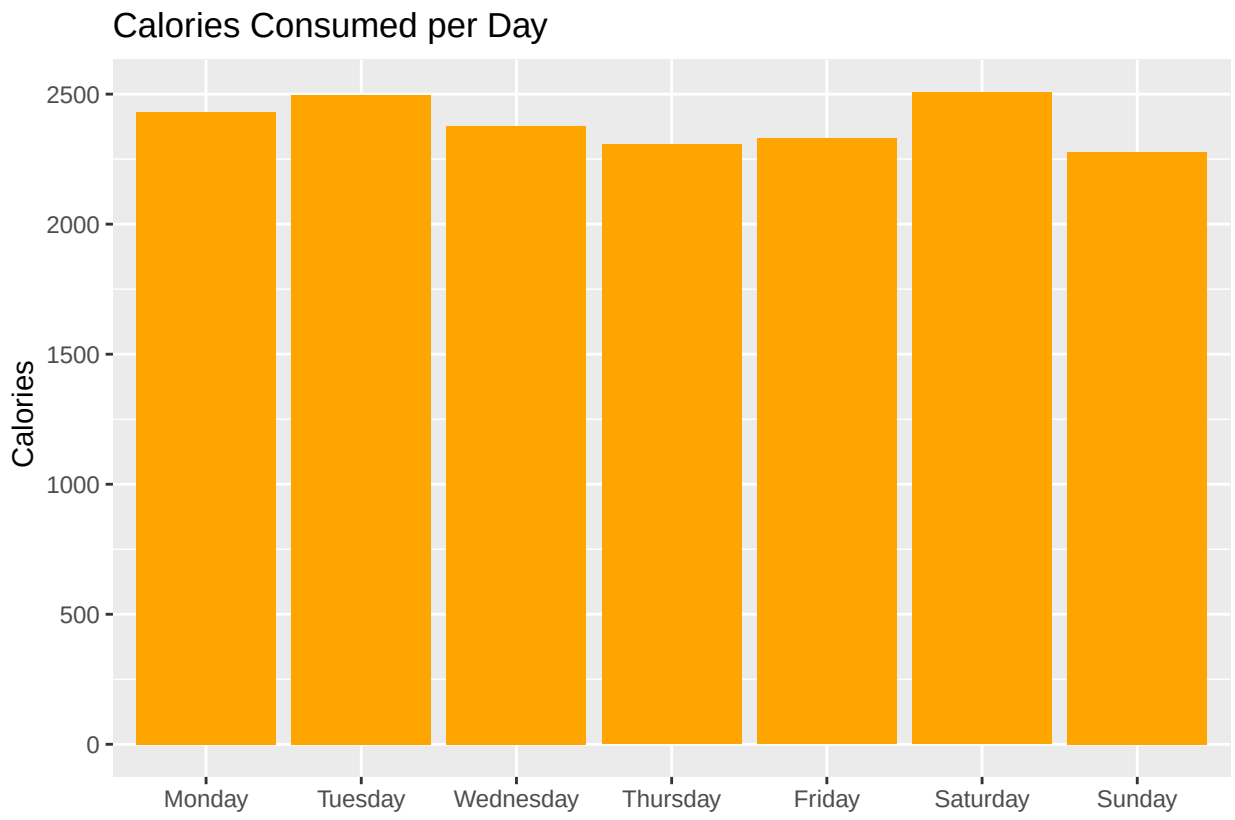

```
# Calculate average hours/minutes of sleep per week

mean(weekday_sleep_summary_group$daily_sleep)/60
```

```
#Number of steps per day of week

ggplot(data = weekday_sleep_summary_group, aes(x = weekday, y = daily_steps)) +
  geom_col(fill = "pink") +
  labs(title = "Steps per Day",
       x = "",
       y = "Steps")
```

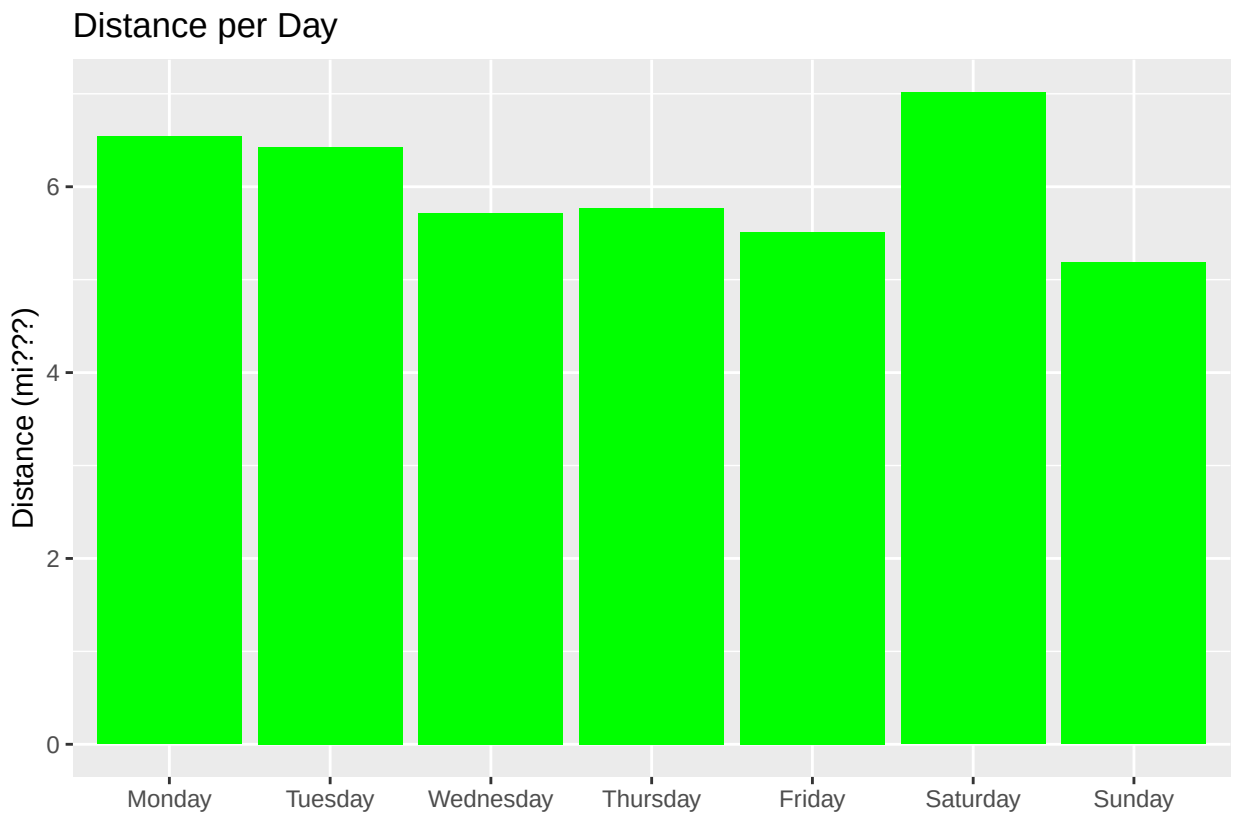



```
# Calculate average steps throughout the week

mean(weekday_sleep_summary_group$daily_steps)
```

```
# Calories per day of week
ggplot(data = weekday_sleep_summary_group, aes(x = weekday, y = daily_calories)) +
  geom_col(fill = "orange") +
  labs(title = "Calories Consumed per Day",
       x = " ",
       y = " Calories")
```

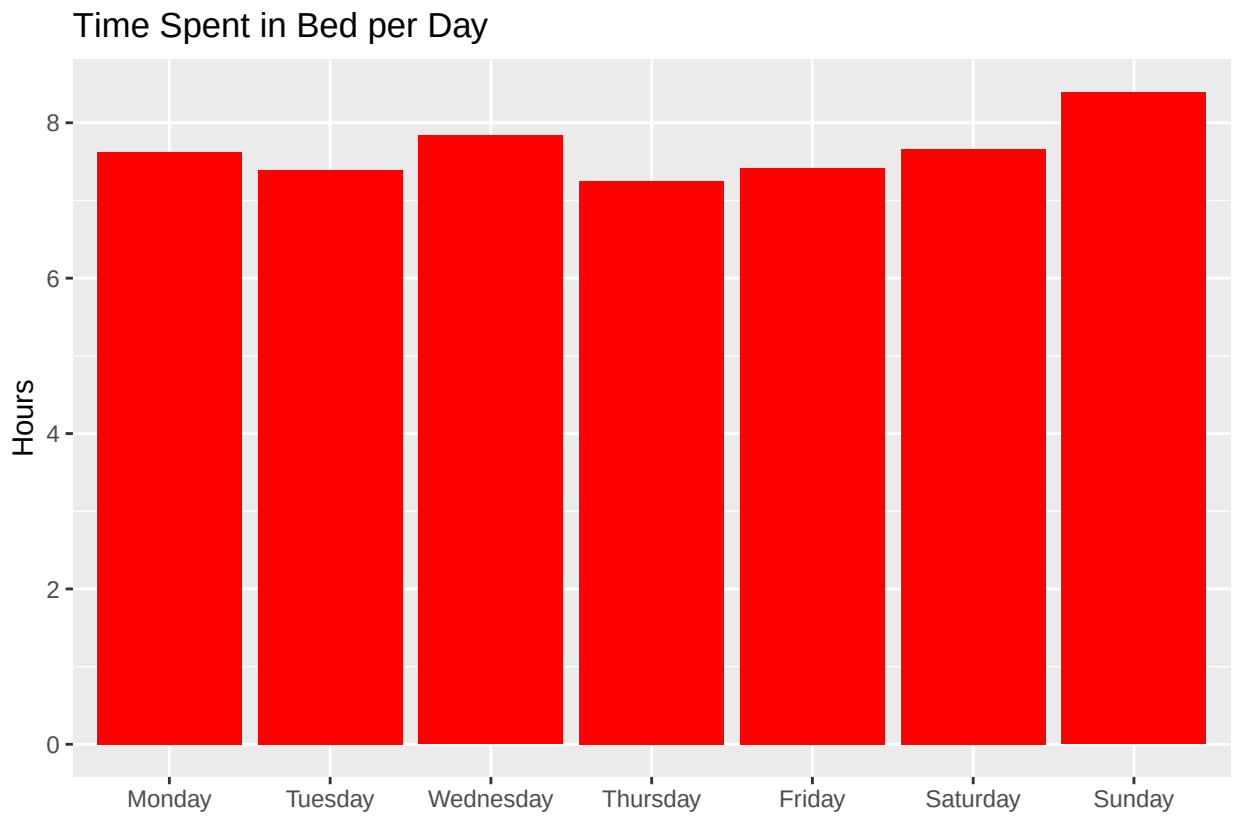



```
# Distance walked per day of week

ggplot(data = weekday_sleep_summary_group, aes(x = weekday, y = daily_distance)) +
  geom_col(fill = "green") +
  labs(title = "Distance per Day",
       x = "",
       y = "Distance (mi???)")
```




```
# Calculate average distance walked throughout the week

mean(weekday_sleep_summary_group$daily_distance)
```

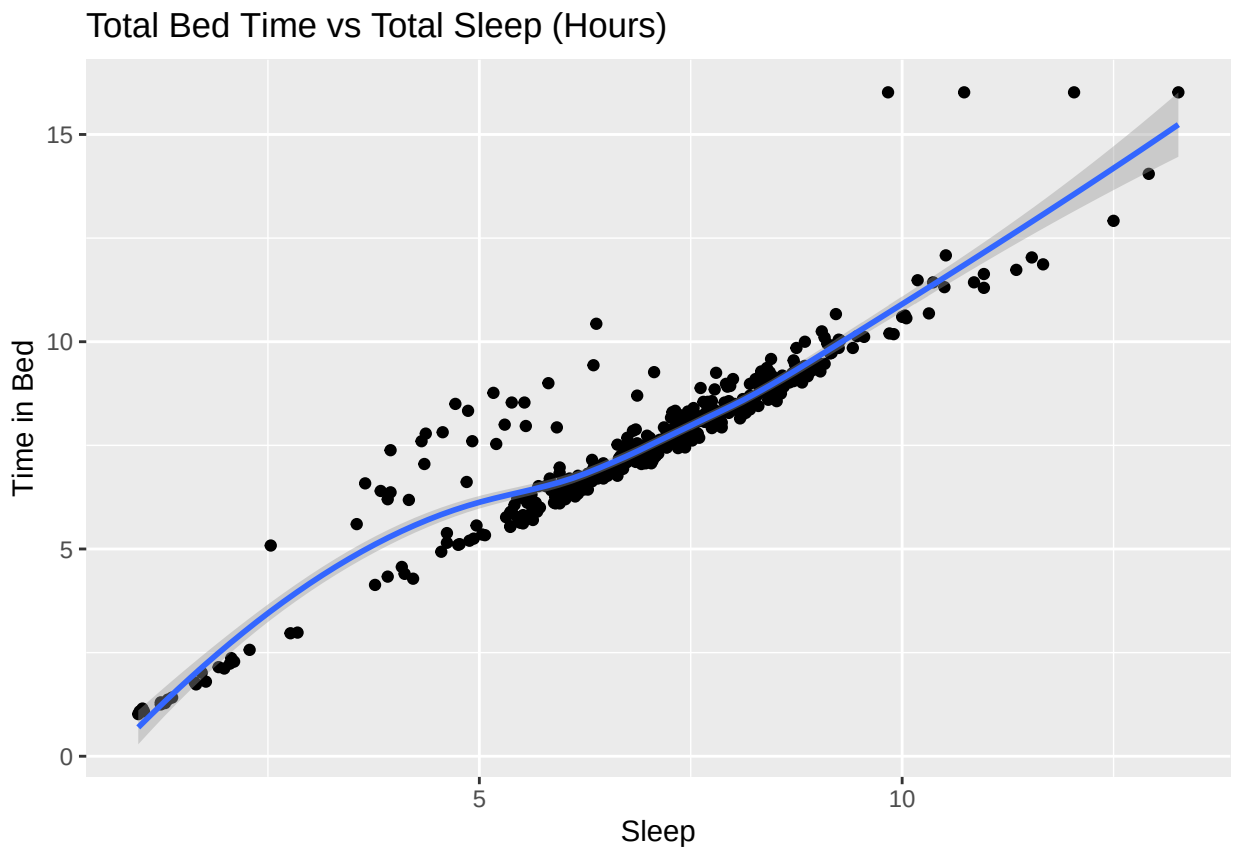
```
# Hours spent in bed per day

ggplot(data = weekday_sleep_summary_group, aes(x = weekday, y = daily_bedtime/60)) +
  geom_col(fill = "red") +
  labs(title = "Time Spent in Bed per Day",
       x = "",
       y = "Hours")
```




```
# Average daily bed time (hours)

mean(weekday_sleep_summary_group$daily_bedtime)/60
```

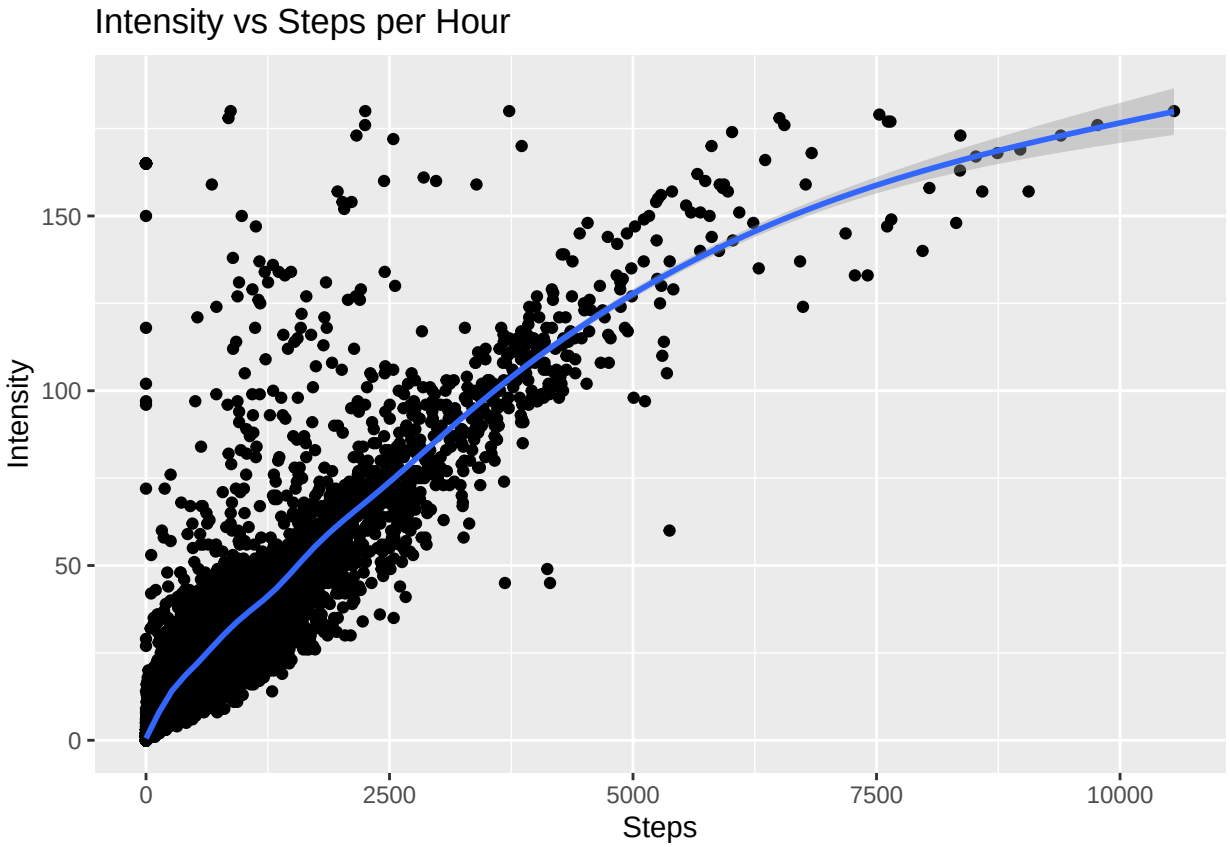
```
# Daily Bed Time vs Sleep - scatterplot

ggplot(data = daily_sleep_activity_merged, aes(x = totalminutesasleep/60, y = totaltimeinbed/60)) +
  geom_point() +
  geom_smooth() +
  labs(title = "Total Bed Time vs Total Sleep (Hours)",
       x = "Sleep",
       y = "Time in Bed")
```

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




```
# Hourly steps vs intensities - scatterplot

ggplot(data = hourly_step_intensities_calories, aes(x = steptotal, y = totalintensity)) +
  geom_point() +
  geom_smooth() +
  labs(title = "Intensity vs Steps per Hour",
       x = "Steps",
       y = "Intensity")
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```




```
# For data sets containing datetime in their "time" referenced columns, split date and time into separa

hourly_intensities <- hourly_intensities %>%
  separate(date_time, into = c("date", "time"), sep = " ")
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 934 rows [1, 25, 49, 73,
## 97, 121, 145, 169, 193, 217, 241, 265, 289, 313, 337, 361, 385, 409, 433, 457,
## ...].
```

```
View(hourly_intensities)
```

```
hourly_steps <- hourly_steps %>%
  separate(date_time, into = c("date", "time"), sep= " ") %>%
  mutate(date = ymd(date))
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 934 rows [1, 25, 49, 73,
## 97, 121, 145, 169, 193, 217, 241, 265, 289, 313, 337, 361, 385, 409, 433, 457,
## ...].
```

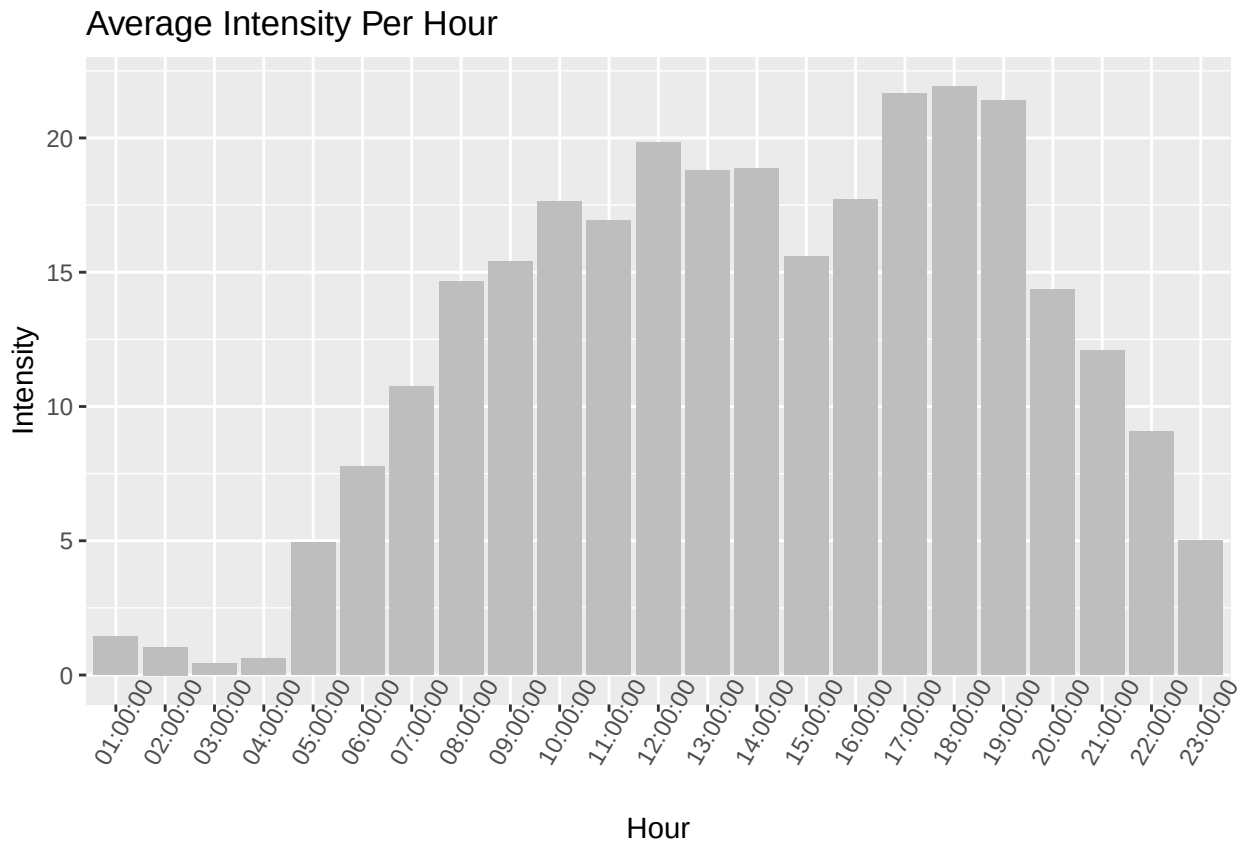
```
View(hourly_steps)
```

```
# Group by new "time"/hour column & drop rows containing missing values (time column will have missing
```

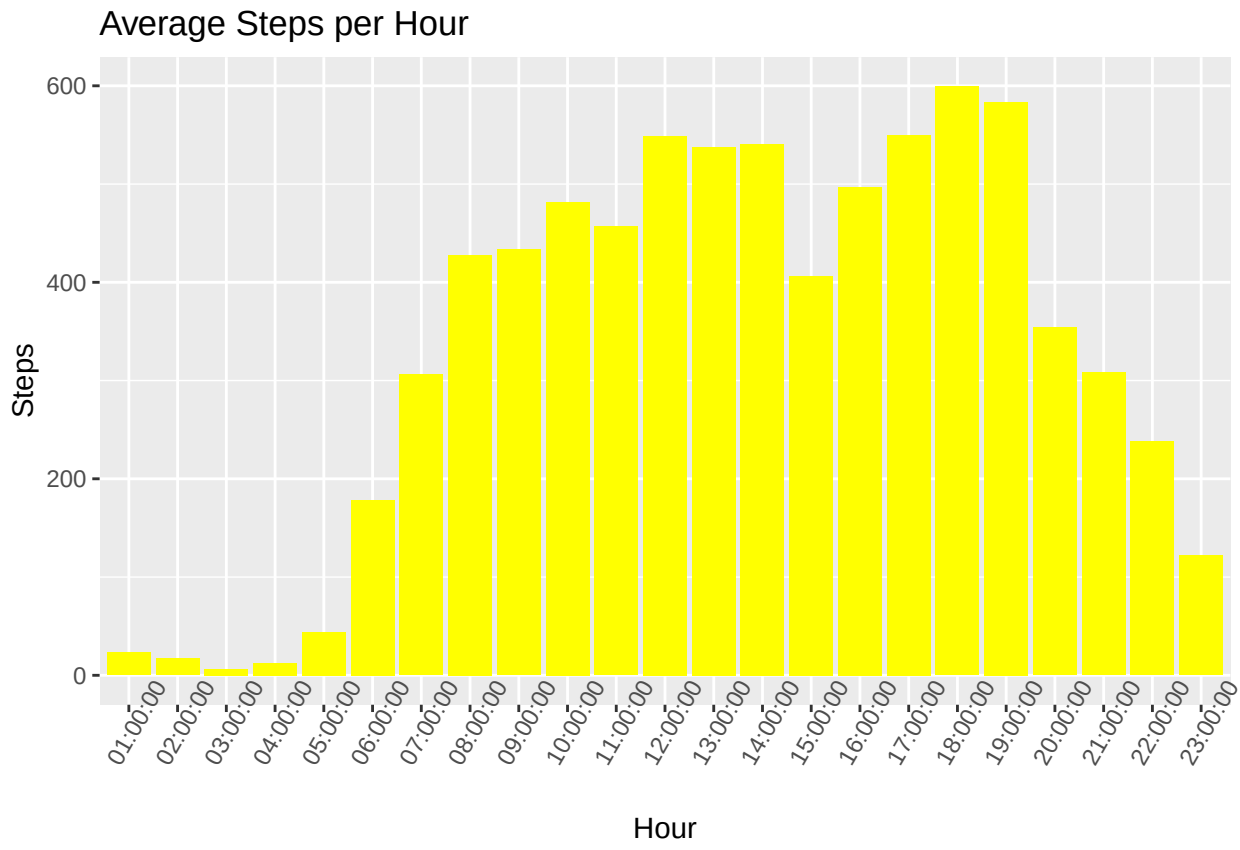
```r
hourly_intensities <- hourly_intensities %>%
  group_by(time) %>%
  drop_na() %>%
  summarise(avg_intensity_per_hr = mean(totalintensity))

View(hourly_intensities)
```

```r
hourly_steps <- hourly_steps %>%
  group_by(time) %>%
  drop_na() %>%
  summarise(avg_steps_per_hr = mean(steptotal))

View(hourly_steps)
```

```r
# Column chart graphing intensity levels by time/hour

ggplot(data = hourly_intensities, aes(x = time, y = avg_intensity_per_hr)) +
  geom_col(fill = "gray") +
  labs(title = "Average Intensity Per Hour",
       x = "Hour",
       y = "Intensity") +
  theme(axis.text.x = element_text(angle = 60))
```



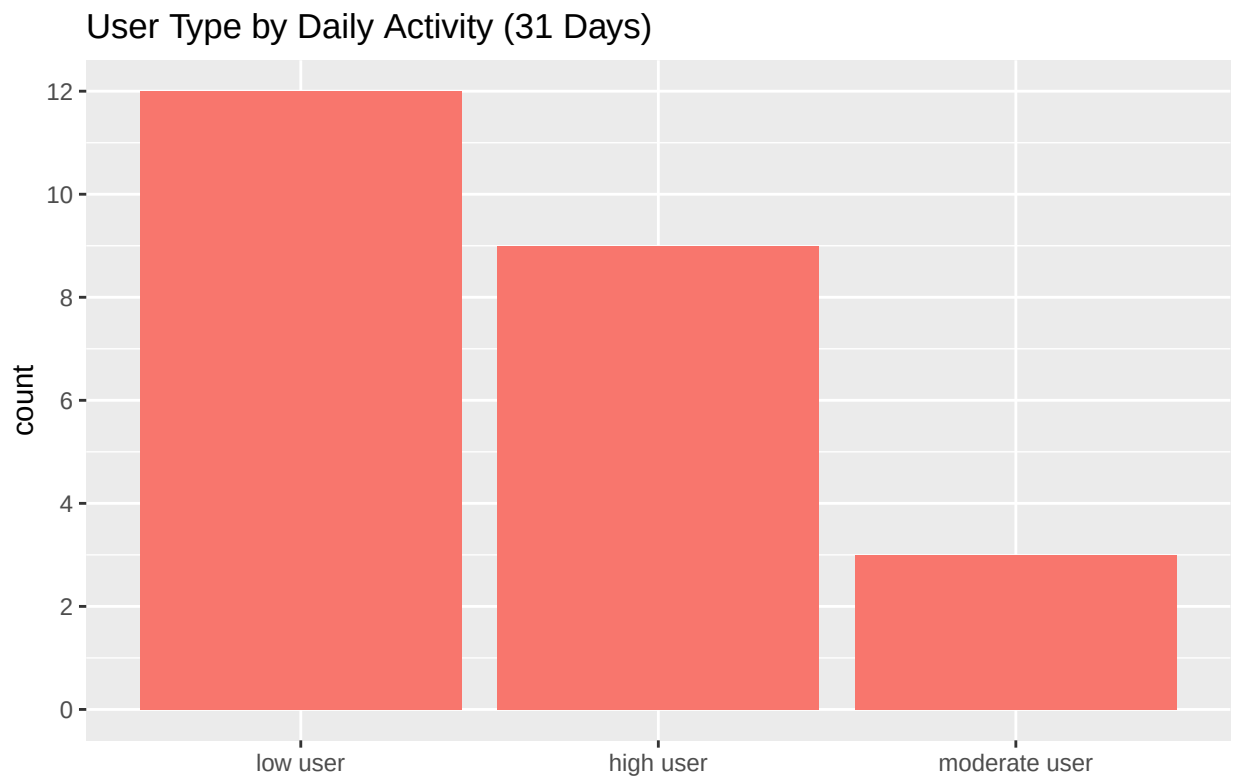
```r
# Column chart graphing steps by time/hour

ggplot(data = hourly_steps, aes(x = time, y = avg_steps_per_hr)) +
  geom_col(fill = "yellow") +
  labs(title = "Average Steps per Hour",
       x = "Hour",
       y = "Steps") +
  theme(axis.text.x = element_text(angle = 60))
```




```r
# Group by and classify "user type" (amount of usage from each user) using "case_when" function

daily_use <- daily_sleep_activity_merged %>%
  group_by(id) %>%
  summarise(days_used = sum(n())) %>%
  mutate(user_type = case_when(
    days_used >=1 & days_used <= 10 ~ "low user",
    days_used >= 11 & days_used <= 20 ~ "moderate user",
    days_used >= 21 & days_used <= 31 ~ "high user"
  ))

View(daily_use)
```

```r
# Bar chart to bucket the number of high, low, and moderate users

ggplot(data = daily_use, aes(x = user_type, fill = "coral")) +
```

```
  geom_bar() +
  scale_x_discrete(labels = c("low user", "high user", "moderate user")) + # "= c(...)" combines values
  scale_y_continuous(breaks = c(0, 2, 4, 6, 8, 10, 12, 14)) +
  labs(title = "User Type by Daily Activity (31 Days)",
       x = "",
       caption = "*Data recorded over a period of 31 days from 4/12/2016 - 5/12/2016") +
  theme(legend.position="none")
```



*Data recorded over a period of 31 days from 4/12/2016 − 5/12/2016

```
# Convert grouped data by user type and set to percentages

daily_use_percent <- daily_use %>%
  group_by(user_type) %>%
  summarise(total = n()) %>%  # n() = number of observations in a group
  mutate(totals = sum(total)) %>%
  group_by(user_type) %>%
  summarise(total_percent = total/totals) %>%
  mutate(labels = scales::percent(total_percent))

View(daily_use_percent)
```

```
# Pie chart from daily_use_percent data

ggplot(data = daily_use_percent, aes(x = " ", y = total_percent, fill = user_type)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +
```

```
  labs(title = "Smart Device Daily Use by User Type",
       caption = "*high user = 21 - 31 days\n moderate user = 11 - 20 days\n low user = 0 - 10 days") +
  theme_minimal() + # gets ride of background coloring
  theme(axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.title.y = element_blank(),
        axis.ticks = element_blank(),
        panel.grid = element_blank()) +
  geom_text(aes(label = labels), # add label to pie chart (% numbers, variable = "labels")
            position = position_stack(vjust = 0.5)) #adjust positioning to center of polygon
```