# Bellabeat Wellness Device Case Study:

# Analyzing User Activity and Wellness

**Project Title**: Bellabeat Wellness Device Case Study: Analyzing User Activity and Wellness Patterns

**Introduction**: The Bellabeat Wellness Device Case Study explores user activity patterns, sleep behavior, and calorie expenditure to gain insights into wellness trends. This analysis was conducted to help Bellabeat, a wellness technology company, understand user behavior and enhance their products’ engagement and impact on health. By leveraging data from Bellabeat devices, this study aims to identify actionable insights to inform product design and marketing strategies.

The study follows a structured 6-step framework, encompassing asking key questions, preparing and processing data, performing analyses, sharing findings through visualizations, and making data-driven recommendations. Each step is documented in detail to demonstrate the analytical process and outcomes.

[1. Ask](#Ask)

[2. Prepare](#Prepare)

[3. Process](#Process)

[4. Analyze](#Analyze)

[5. Share](#Share)

[6. Act](#Act)

**1. Ask**

* **Objective**: Define the business task and set clear objectives for the analysis.
* **Business Task**: Analyze activity and wellness data from Bellabeat devices to understand user behavior and trends. The goal is to identify actionable insights that can guide product improvements and marketing strategies for Bellabeat’s wellness products.
* **Questions**:
  + What are the daily activity patterns of Bellabeat users?
  + How do steps, calories, and sleep patterns vary across the week?
  + Are there any correlations between steps taken and calories burned?

**2. Prepare**

* **Data Sources**: [Bellabeat wellness data,](https://www.kaggle.com/datasets/arashnic/fitbit" \t "_blank) which includes metrics such as daily steps, calories burned, activity levels, and sleep duration.
* **Setup and Data Import**

# Install necessary packages and load libraries

install.packages("tidyverse")

library(tidyverse)

library(lubridate)

# Load datasets

daily\_activity <- read\_csv("dailyActivity\_all.csv")

daily\_calories <- read\_csv("daily\_calories\_aggr.csv")

daily\_sleep <- read\_csv("daily\_total\_sleep\_aggr.csv")

daily\_steps <- read\_csv("distinct\_daily\_steps.csv")

* **Initial Data Inspection**

# Check column names and sample data

colnames(daily\_activity)

colnames(daily\_calories)

colnames(daily\_sleep)

colnames(daily\_steps)

head(daily\_activity)

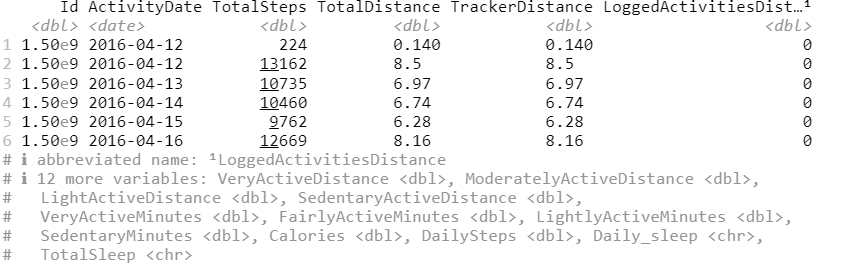
head(daily\_calories)

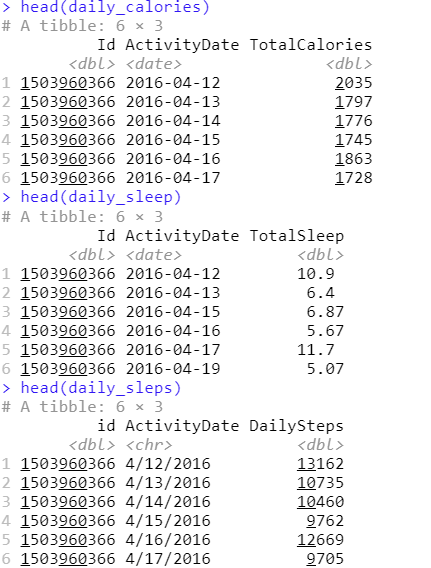
head(daily\_sleep)

head(daily\_steps)

A screenshot of a computer

Description automatically generated





* **Data Preparation: Rename Columns**

library(dplyr)

# Convert column names to CamelCase for consistency

daily\_calories <- daily\_calories %>% rename(Id = id , TotalCalories = total\_calories)

daily\_sleep <- daily\_sleep %>% rename(Id = id , TotalSleep = total\_sleep)

daily\_steps <- daily\_steps %>% rename(Id = id)

* **Data Preparation: Date Format Conversion**

# Standardize date formats

daily\_activity$ActivityDate <- mdy(daily\_activity$ActivityDate)

daily\_steps$ActivityDate <- mdy(daily\_steps$ActivityDate)

daily\_sleep$ActivityDate <- ymd(daily\_sleep$ActivityDate)

daily\_calories$ActivityDate <- ymd(daily\_calories$ActivityDate)

**3. Process**

* **Duplicate Check and Handling**

# Check for duplicates

duplicates <- daily\_activity %>%

  group\_by(Id, ActivityDate) %>%

  filter(n() > 1)

print(duplicates)

View(duplicates)

* **Missing Values Handling**

# Replace N/A values in DailySleep with 0

daily\_activity <- daily\_activity %>%

  mutate(TotalSleep = ifelse(Daily\_sleep == "#N/A", 0 , Daily\_sleep))

* **Data Aggregation**

# Summarize data by Id and ActivityDate

daily\_activity\_cleaned <- daily\_activity %>%

  group\_by(Id, ActivityDate) %>%

  summarise(

    TotalSteps = sum(TotalSteps, na.rm = TRUE),

    TotalDistance = sum(TotalDistance, na.rm = TRUE),

    TrackerDistance = sum(TrackerDistance, na.rm = TRUE),

    VeryActiveDistance = sum(VeryActiveDistance, na.rm = TRUE),

    ModeratelyActiveDistance = sum(ModeratelyActiveDistance, na.rm = TRUE),

    LightActiveDistance = sum(LightActiveDistance, na.rm = TRUE),

    VeryActiveMinutes = max(VeryActiveMinutes, na.rm = TRUE),

    FairlyActiveMinutes = max(FairlyActiveMinutes, na.rm = TRUE),

    LightlyActiveMinutes = max(LightlyActiveMinutes, na.rm = TRUE),

    SedentaryMinutes = max(SedentaryMinutes, na.rm = TRUE),

    Calories = max(Calories, na.rm = TRUE),

    DailySteps = max(DailySteps, na.rm = TRUE),

    DailySleep = max(TotalSleep, na.rm = TRUE),

    .groups = "drop"

  )

# View the cleaned data

View(daily\_activity\_cleaned)

head(daily\_activity\_cleaned)

colnames(daily\_activity\_cleaned)

A screenshot of a computer

Description automatically generated

* **Check for Missing Values**

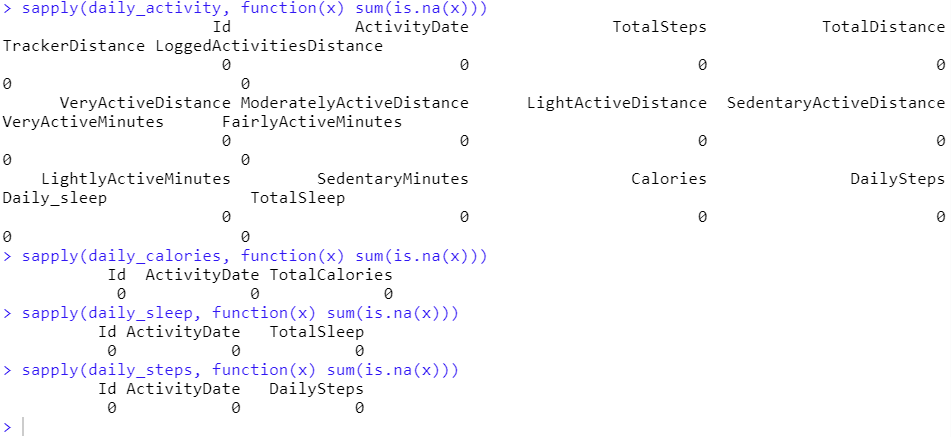
# Check for missing values in each dataset

sapply(daily\_activity, function(x) sum(is.na(x)))

sapply(daily\_calories, function(x) sum(is.na(x)))

sapply(daily\_sleep, function(x) sum(is.na(x)))

sapply(daily\_steps, function(x) sum(is.na(x)))



**4. Analyze**

* **Summary Statistics**

# Calculate summary statistics

summary\_data <- daily\_activity\_cleaned %>%

  summarise(

    min\_steps = min(DailySteps, na.rm = TRUE),

    max\_steps = max(DailySteps, na.rm = TRUE),

    avg\_steps = mean(DailySteps, na.rm = TRUE),

    min\_calories = min(Calories, na.rm = TRUE),

    max\_calories = max(Calories, na.rm = TRUE),

    avg\_calories = mean(Calories, na.rm = TRUE),

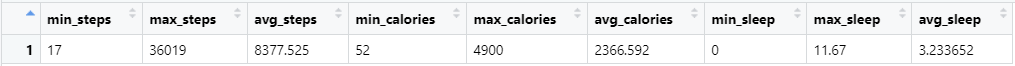
    min\_sleep = min(DailySleep , na.rm = TRUE),

    max\_sleep = max(DailySleep , na.rm = TRUE),

    avg\_sleep = mean(DailySleep , na.rm = TRUE)

  )

View(summary\_data)



* **Step and Calorie Distribution**

# Create step and calorie ranges

step\_distribution <- daily\_activity\_cleaned %>%

  mutate(

    step\_range = case\_when(

      DailySteps < 5000 ~ "(1) <5000",

      DailySteps >= 5000 & DailySteps <= 10000 ~ "(2) 5000 - 10000",

      DailySteps >= 10001 & DailySteps <= 15000 ~ "(3) 10001 - 15000",

      DailySteps > 15000 ~ "(4) >15000"

    )

  ) %>%

  group\_by(step\_range) %>%

  summarise(count\_days = n()) %>%

  arrange(step\_range)

calorie\_distribution <- daily\_activity\_cleaned %>%

  mutate(

    calorie\_range = case\_when(

      Calories < 1000 ~ "(1) <1000",

      Calories >= 1000 & Calories <= 2000 ~ "(2) 1000-2000",

      Calories > 2000 ~ "(3) >2000"

    )

  ) %>%

  group\_by(calorie\_range) %>%

  summarise(count\_days = n()) %>%

  arrange(calorie\_range)

# Plot the step distribution

library(ggplot2)

ggplot(step\_distribution, aes(x = step\_range, y = count\_days, fill = step\_range)) +

  geom\_bar(stat = "identity") +

  labs(title = "Distribution of Daily Steps",

       x = "Step Range",

       y = "Count of Days") +

  theme\_minimal() +

  theme(legend.position = "none")

# Plot the calorie distribution

ggplot(calorie\_distribution, aes(x = calorie\_range, y = count\_days, fill = calorie\_range)) +

  geom\_bar(stat = "identity") +

  labs(

    title = "Calorie Distribution by Range",

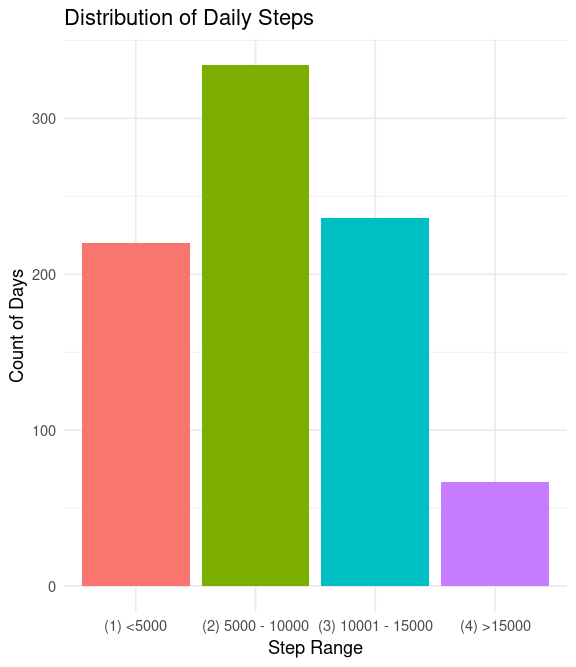
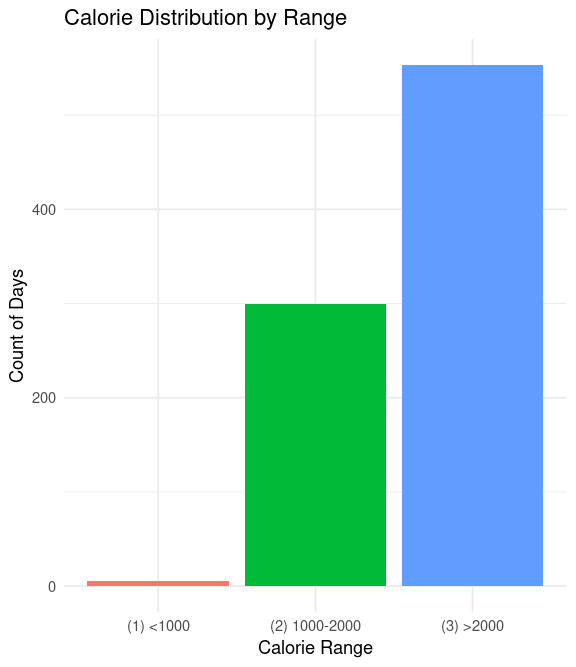
    x = "Calorie Range",

    y = "Count of Days"

  ) +

  theme\_minimal() +

  theme(legend.position = "none")



* **Daily Summary Over Time**

# Average daily steps and calories over time

daily\_summary <- daily\_activity\_cleaned %>%

  group\_by(ActivityDate) %>%

  summarise(

    avg\_steps = mean(DailySteps,na.rm = TRUE),

    avg\_calories = round(mean(Calories,na.rm = TRUE),2)) %>%

  arrange(ActivityDate)

# Plot daily\_summary data

ggplot(daily\_summary, aes(x = ActivityDate)) +

  geom\_line(aes(y = avg\_steps, color = "Average Steps"), size = 1) +

  geom\_line(aes(y = avg\_calories, color = "Average Calories"), size = 1) +

  labs(

    title = "Average Daily Steps and Calories Over Time",

    x = "Date",

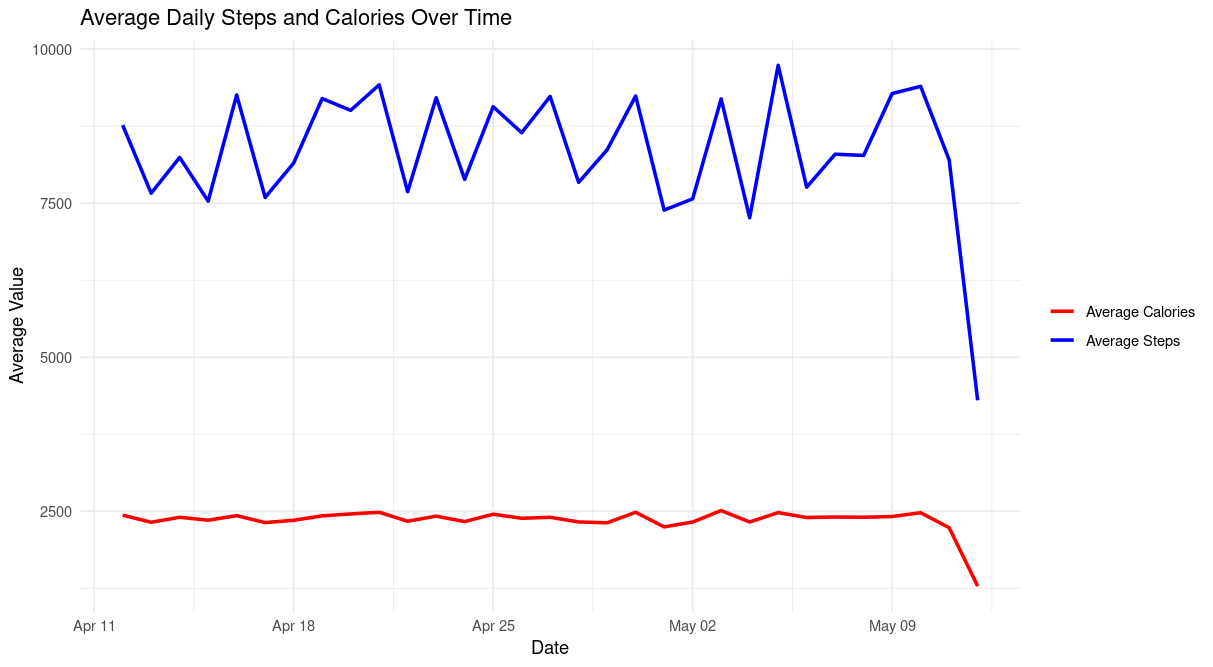
    y = "Average Value"

  ) +

  scale\_color\_manual(values = c("Average Steps" = "blue", "Average Calories" = "red")) +

  theme\_minimal() +

  theme(legend.title = element\_blank())



* **Weekly Trends**

# Add day of week column

daily\_activity\_cleaned <- daily\_activity\_cleaned %>%

  mutate(DayOfWeek = wday(ActivityDate, label = TRUE))

# Weekly summary

weekly\_summary <- daily\_activity\_cleaned %>%

  group\_by(DayOfWeek) %>%

  summarise(

    avg\_steps = mean(DailySteps, na.rm = TRUE),

    avg\_calories = mean(Calories, na.rm = TRUE)

  )

# Plot for weekly trends

ggplot(weekly\_summary, aes(x = DayOfWeek, y = avg\_steps, fill = avg\_steps)) +

  geom\_bar(stat = "identity") +

  scale\_fill\_gradient(low = "lightblue", high = "blue") +

  labs(title = "Average Steps by Day of the Week", x = "Day of the Week", y = "Average Steps") +

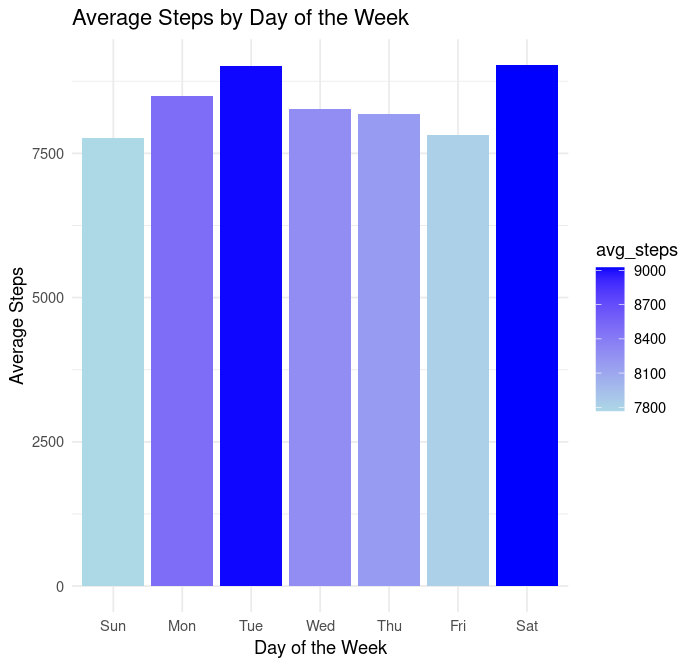
  theme\_minimal()

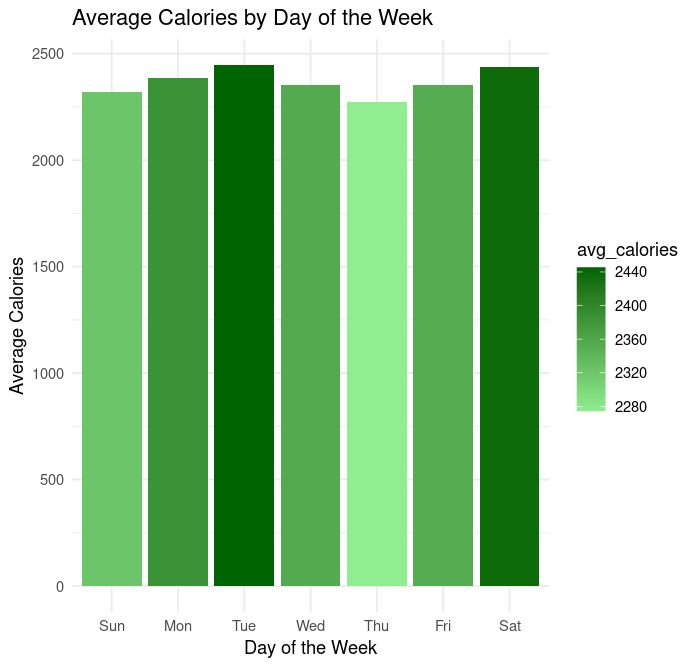
ggplot(weekly\_summary, aes(x = DayOfWeek, y = avg\_calories, fill = avg\_calories)) +

  geom\_bar(stat = "identity") +

  scale\_fill\_gradient(low = "lightgreen", high = "darkgreen") +

  labs(title = "Average Calories by Day of the Week", x = "Day of the Week", y = "Average Calories") +

  theme\_minimal()



* **Weekly Patterns in Activity and Movement**

# Summarize the data by DayOfWeek

weekly\_activity\_summary <- daily\_activity\_cleaned %>%

  group\_by(DayOfWeek) %>%

  summarise(

    avg\_very\_active = mean(VeryActiveMinutes, na.rm = TRUE),

    avg\_light\_active = mean(LightlyActiveMinutes, na.rm = TRUE),

    avg\_sedentary = mean(SedentaryMinutes, na.rm = TRUE),

    avg\_very\_active\_distance = mean(VeryActiveDistance, na.rm = TRUE),

    avg\_moderately\_active\_distance = mean(ModeratelyActiveDistance, na.rm = TRUE),

    avg\_light\_active\_distance = mean(LightActiveDistance, na.rm = TRUE)

  )

# Reshape the data into long format for plotting

weekly\_activity\_long <- weekly\_activity\_summary %>%

  pivot\_longer(cols = starts\_with("avg\_"),

               names\_to = "ActivityType",

               values\_to = "Minutes")

# Plot weekly activities

ggplot(weekly\_activity\_long, aes(x = DayOfWeek, y = Minutes, fill = ActivityType)) +

  geom\_col(position = "dodge") +

  facet\_wrap(~ ActivityType, scales = "free\_y") +

  labs(title = "Distribution of Activity Levels Across the Week",

       x = "Day of the Week",

       y = "Average Minutes") +

  scale\_fill\_manual(values = c("avg\_very\_active" = "darkgreen", "avg\_light\_active" = "lightgreen", "avg\_sedentary" = "orange", "avg\_very\_active\_distance" = "darkblue", "avg\_moderately\_active\_distance" = "lightblue", "avg\_light\_active\_distance" = "yellow")) +

  theme\_minimal()

A group of different colored bars

Description automatically generated

* **Outlier Detection**

# Detect outliers

mean\_steps <- mean(daily\_activity\_cleaned$DailySteps, na.rm = TRUE)

sd\_steps <- sd(daily\_activity\_cleaned$DailySteps, na.rm = TRUE)

outliers <- daily\_activity\_cleaned %>%

  filter(DailySteps > (mean\_steps + 2 \* sd\_steps) | DailySteps < 1000)

outliers <- outliers %>%

  mutate(avg\_daily\_steps = mean\_steps,

         standard\_deviation = sd\_steps)

# Plot outliers

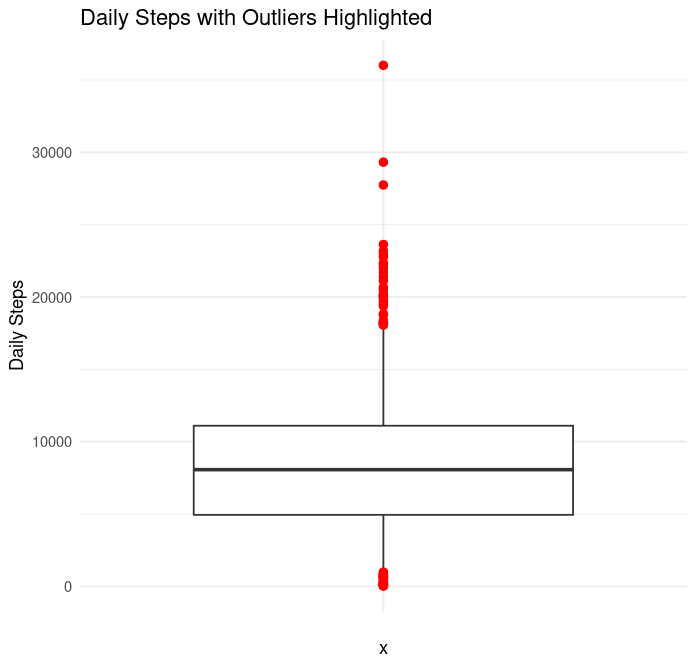
ggplot(daily\_activity\_cleaned, aes(x = "", y = DailySteps)) +

  geom\_boxplot(outlier.shape = NA) +

  geom\_point(data = outliers, aes(y = DailySteps), color = "red", size = 2) +

  labs(title = "Daily Steps with Outliers Highlighted", y = "Daily Steps") +

  theme\_minimal()



* **Correlation Analysis**

# Correlation

correlation <- cor(daily\_activity\_cleaned$DailySteps, daily\_activity\_cleaned$Calories, use = 'complete.obs')

ggplot(daily\_activity\_cleaned, aes(x = DailySteps, y = Calories)) +

  geom\_point(color = 'blue', alpha = 0.5) +

  geom\_smooth(method = 'lm', se = FALSE, color = 'red') +

  labs(

    title = 'Daily Steps vs. Total Calories Burned',

    x = 'Daily Steps',

    y = 'Total Calories Burned'

  ) +

  theme\_minimal()

**A graph showing a red line and blue dots

Description automatically generated**

**5. Share**

**Key Analysis Steps and Findings:**

1. **Summary Statistics**:
   * Calculated key statistics, including minimum, maximum, and average values for daily steps, calories burned, and sleep duration.
   * These summary metrics help provide a quick overview of the data and set a foundation for deeper analysis.
2. **Activity and Calorie Distribution**:
   * Categorized daily steps into ranges (<5000, 5000-10000, 10001-15000, >15000) and visualized the distribution to understand user activity levels.
   * A similar approach was used for calorie ranges, providing insight into users’ daily energy expenditure.
3. **Trends Over Time**:
   * Analyzed daily steps and calories over time to observe any patterns or fluctuations in activity.
   * Created visualizations to show weekly trends, which indicated patterns in user activity that vary by day of the week.
4. **Correlation Analysis**:
   * Calculated the correlation between daily steps and calories to assess the relationship between physical activity and calorie expenditure.
   * A moderate positive correlation was found, suggesting that as steps increase, calorie burn generally increases as well.
5. **Outlier Detection**:
   * Identified days with unusually high or low activity levels using statistical thresholds.
   * Highlighted these outliers in visualizations to help Bellabeat identify opportunities for user engagement.

**6. Act**

**Recommendations for Bellabeat**

Based on the insights gathered from the analysis, the following recommendations can help Bellabeat enhance user engagement and promote healthier lifestyles:

**1. Introduce Activity Challenges and Reminders on Weekends:**

Analysis showed that users tend to be more sedentary during the weekends. Bellabeat can introduce weekend challenges or send reminders to encourage users to increase their activity levels.

**2.** **Promote Notifications for Low Activity or Calorie Burn Days**

Bellabeat could implement notifications for users who have low daily steps or calorie burn, encouraging them to stay active. Personalized messages could remind users of their health goals and suggest achievable activities to help them stay on track. This would create a more supportive experience and help users develop consistent wellness habits.

**Next Steps**

To further refine Bellabeat’s offerings, consider these next steps:

* **User Segmentation Analysis**: Identify distinct user segments based on activity, calorie burn, and sleep patterns. This will enable Bellabeat to offer more personalized recommendations and tailor marketing strategies to different user needs.
* **Enhanced Sleep Insights**: Explore the relationship between sleep patterns and activity levels to offer advice on sleep quality improvements, aligning with holistic wellness goals.

**Tools and Skills Demonstrated**

**R Programming**: Used extensively for data cleaning, transformation, and statistical analysis. Leveraged libraries such as dplyr and tidyverse for efficient data manipulation, and ggplot2 for visualization.

**Data Cleaning**: Ensured data consistency and accuracy by handling missing values, removing duplicates, and converting data types. Utilized conditional transformations to prepare datasets for analysis.

**Data Analysis**: Conducted Exploratory Data Analysis (EDA) to examine distributions, detect outliers, and calculate correlations between key metrics (daily steps and calorie burn).

**Data Visualization**: Created informative and visually engaging plots using ggplot2 to illustrate weekly activity patterns, distribution of steps and calories, and correlations. Faceted and gradient-filled charts enabled clear communication of insights.

**Conclusion**

This analysis of Bellabeat’s wellness data highlights the potential for enhancing user engagement through data-driven insights. By identifying trends in user activity, sleep, and calorie expenditure, Bellabeat can introduce personalized features that align with users’ wellness goals.