WELLNESS TECHNOLOGY COMPANY CASE STUDY

As part of my Google Data Analytics Certification, I conducted an in-depth analysis of smart-device fitness tracking data to uncover insights into user wellness behaviors for Bellabeat, a wellness technology company. Using R for data analysis and visualization, I explored weekly activity patterns, sedentary behaviors, and correlations between activity levels, calorie expenditure, and sedentary time.

This case study analyzes wellness behaviors of FitBit users to provide clear, actionable insights for Bellabeat. Results revealed peak activity on Mondays and Wednesdays, with notable declines on weekends. Sedentary behavior was higher toward the week's end. Recommendations include personalized activity notifications for less active days, targeted features in the Bellabeat Leaf to reduce sedentary time, and strategic marketing aligned with peak activity periods.

This case study highlights my skills in preparing and analyzing real-world data, identifying actionable trends, and translating analytical insights into strategic business recommendations.

1. ASK - A Clear Statement Of The Business Task

Bellabeat seeks actionable insights derived from the analysis of publicly available fitness-tracking data to better understand consumer behaviors around wellness and hydration. Specifically, this case study aims to uncover meaningful trends from FitBit device usage data that can inform strategic marketing initiatives for Bellabeat’s Leaf wellness tracker.

Objectives:

- Identify key patterns and trends in users’ fitness and wellness behaviors, focusing particularly on activity levels, sleep patterns, and sedentary habits;
- Apply these insights to anticipate how Bellabeat’s target demographic might engage with the Leaf wellness tracker in their daily routines;
- Develop targeted, data-driven marketing recommendations to enhance consumer engagement, drive product adoption, and position the Bellabeat Leaf strategically within the competitive wellness technology market.

Key stakeholders:

- Urška Sršen (Bellabeat Co-founder & Chief Creative Officer);
- Bellabeat Marketing Analytics Team;
- Potential and existing Bellabeat customers interested in health and wellness products.

2. PREPARE - A Description Of All Data Sources Used

Data Sources Used

For this case study, we are analyzing smart device usage data to identify wellness and activity trends and provide marketing insights for Bellabeat's Leaf wellness tracker. The dataset used comes from a publicly available fitness tracker dataset that contains data collected from FitBit users.

1. Primary Data Source

- **Dataset Name:** FitBit Fitness Tracker Data
- **Source:** Kaggle - FitBit Fitness Tracker Data
- **License:** CC0: Public Domain (freely available for public use)
- **Storage:** The dataset is stored locally, organized into two folders:
    - **Folder 1:** Data collected from December 3 to 4, 2016 (11 CSV files).
    - **Folder 2:** Data collected from December 4 to 5, 2016 (18 CSV files).
- **Format:** Each file is in CSV format, with wide format for summary data and long format for minute-level tracking data.

2. Data description

This dataset consists of multiple CSV files, each containing data related to smart device usage, fitness activities, and health monitoring. The primary data points include:

File Name	Description	Data Format
dailyActivity.csv	Summary of daily activity (steps, calories, sedentary time).	Wide format
hourlySteps.csv	Number of steps taken per hour.	Wide format
hourlyCalories.csv	Calories burned per hour.	Wide format
minuteSleep.csv	Sleep data at minute-level.	Long format
sleepDay.csv	Summary of total sleep per day.	Wide format
dailyIntensities.csv	Activity levels (sedentary, lightly active, very active).	Wide format
heartRateSeconds.csv	Heart rate measured per second.	Long format
weightLogInfo.csv	Users' logged weight records.	Wide format
waterIntake.csv (if available)	Water consumption data.	Wide format

3. Bias, Credibility and Data Integrity

Bias and Credibility Analysis

Using the **ROCCC Framework** (Reliable, Original, Comprehensive, Current, Cited):

- **Reliable:** The dataset is from Kaggle, a trusted source, but collected from volunteer users.
- **Original:** No, this is third-party data collected by FitBit users, not directly from Bellabeat.
- **Comprehensive:** Partially. It covers various fitness metrics, but only for 30 users.

- **Current:** No, the dataset is from 2016, which may not fully reflect current smart device trends.
- **Cited:** Yes, the dataset source (Mobius on Kaggle) is clearly referenced.

Key Concerns:

1. **Small Sample Size** - Only 30 users, which is not representative of the entire smart device market.
2. **Self-Selection Bias** - Users opted in, meaning they may be more health-conscious than the general population.
3. **Device-Specific Bias** - Data comes from FitBit devices, so it may not fully represent Bellabeat users.

4. Licensing, Privacy, and Security Considerations

- **Licensing:**
    - This dataset is under CC0: Public Domain, meaning it is free to use without restrictions.
- **Privacy & Security:**
    - The dataset does not contain personally identifiable information (PII).
    - User IDs are anonymized, ensuring privacy protection.
- **Accessibility:**
    - The dataset is freely available on Kaggle, making it easy to access and use.

Conclusion: There are no major privacy concerns, and the dataset is safe to use.

5. Data Integrity Verification

To ensure data integrity, the following checks will be conducted in **R**:

1. **Checking for Missing Data**
   sum(is.na(df)) - This ensures that we identify any NULL or missing values**.**
2. **Checking for Duplicates**
   df %>% distinct() %>% nrow() - This ensures that there are no duplicate records**.**
3. **Checking Data Types**
   str(df) - Confirms that all columns have the correct data type (e.g., numeric, datetime).
4. **Timestamp Verification**
   range(df$timestamp) - Ensures timestamps fall within the expected range**.**
5. **Outlier Detection**
   boxplot(df$water_intake) - Identifies any extreme values that may indicate data errors**.**

Conclusion: These checks will ensure data consistency and reliability before analysis.

Relevance to Case Study

Our goal is to analyze wellness and activity trends and use those insights to improve marketing strategies for Bellabeat's Leaf wellness tracker.

How this dataset helps answer the research question:

1. **Smart Device Usage Trends:**
   - Helps understand **how** users interact with fitness and health tracking devices.
2. **Activity & Sedentary Behaviour Patterns:**
   - We can analyze whether higher activity levels correlate with reduced sedentary time and better wellness habits.
3. **Marketing Strategy Development:**
   - Understanding these behaviors helps target the right audience for the Bellabeat Leaf.

Challenges:

1. **No Direct Device Match** - The dataset comes from FitBit users, not Bellabeat devices, which may introduce product-specific bias.
2. **Short Timeframe (2 Days)** - Limited ability to analyze wellness behaviour patterns.

Conclusion: Even with limitations, the dataset provides valuable insights into smart device usage, activity levels, and wellness behavior, which can inform strategic positioning of the Bellabeat Leaf.

3. PROCESS - Documentation Of Any Cleaning Or Manipulation Of Data

1. What tools are you choosing and why?

I selected R as the primary analytical tool because it offers robust and flexible libraries suitable for data cleaning, manipulation, statistical analysis, and visualization. Specifically, I used:

- *tidyverse* for comprehensive data manipulation, cleaning, and transformation.
- *readr* for importing CSV files efficiently.
- *lubridate* for managing date-time variables seamlessly.

2. Have you ensured your data's integrity?

Yes, thorough integrity checks were conducted at multiple stages, including verifying the completeness of data, consistency in formats, identifying duplicates, and detecting outliers.

3. What steps have you taken to ensure that your data is clean?

The cleaning process included:

- Identifying and handling missing values.
- Checking and removing duplicates.
- Validating and correcting data types.
- Detecting and managing outliers.

4. How can you verify that your data is clean and ready to analyze?

Post-cleaning validation involved:

- Reconfirming the absence of missing or duplicated data.
- Ensuring consistency in data types.
- Reviewing data distributions through summary statistics and visualizations (boxplots).

5. Have you documented your cleaning process so you can review and share those results?

Yes, each step was meticulously documented, including the R scripts used, issues encountered, solutions applied, and the final clean dataset exported for further analysis.

Key Tasks and Detailed Documentation:

Step 1: Install & Load Necessary Packages

```
install.packages("tidyverse")
install.packages("readr")
install.packages("lubridate")
library(tidyverse)  # For data manipulation
library(readr)      # For reading CSV files
library(lubridate)  # For handling dates and times
```

Step 2: **Set Your Folder Paths**

```
folder_1_path <- "/C:/Users/Carmen/Desktop/R/1"  # First folder (11 CSV files)
folder_2_path <- "/C:/Users/Carmen/Desktop/R/2"  # Second folder (18 CSV files)
```

Step 3: Load a Single CSV File (Test Run)

To ensure everything is working, i listed all files in one folder and load one file

1. **List available files in the first folder**

```
list.files(folder_1_path) # This will return a list of all CSV files inside /C:/Users/Carmen/Desktop/R/1
```

2. **Load one file to check its structure:**

```
# Example: Load the "dailyActivity_merged.csv" file from folder 1
df <- read_csv(file.path(folder_1_path, "dailyActivity_merged.csv"))
head(df)
```

Step 4: Load All CSV Files at Once

Since i have multiple files, i will load them automatically from both folders.

```
# Function to read all CSV files from a given folder
load_all_csv <- function(folder_path) {
  files <- list.files(folder_path, pattern = "\\.csv$", full.names = TRUE)  # Get CSV files
  data_list <- lapply(files, read_csv)  # Read each file
  names(data_list) <- basename(files)   # Name them based on file names
  return(data_list)
}
# Load datasets from both folders
data_folder_1 <- load_all_csv(folder_1_path)  # 11 CSV files
data_folder_2 <- load_all_csv(folder_2_path)  # 18 CSV files
# Check loaded file names
names(data_folder_1)
names(data_folder_2)
```

This will load all CSV files into two lists:

- data_folder_1 contains 11 datasets from /C:/Users/Carmen/Desktop/R/1
- data_folder_2 contains 18 datasets from /C:/Users/Carmen/Desktop/R/2

Step 5: Check for Missing Values

```
# Function to count missing values in each dataset
check_missing_values <- function(data_list) {
  sapply(data_list, function(df) sum(is.na(df)))  # Count missing values in each file
}
# Run missing value check
check_missing_values(data_folder_1)
check_missing_values(data_folder_2)
```

The output shows 31 missing values for *weightLogInfo_merged.csv* in 1st folder, and 65 missing values for *weightLogInfo_merged.csv* in 2nd folder.

Missing values were removed using::

```
df <- na.omit(df)  # Remove rows with missing values
```

Result: 457 obs. of 15 variables

```
View(df)
```

Step 6: Check for Duplicate Rows

```
# Function to check for duplicates in each dataset
check_duplicates <- function(data_list) {
  sapply(data_list, function(df) sum(duplicated(df)))
}
# Run duplicate check
check_duplicates(data_folder_1)
check_duplicates(data_folder_2)
```

```
  names(data_folder_1)
  [1] "dailyActivity_merged.csv"                 - 0 duplicates
  [2] "heartrate_seconds_merged.csv"             - 0 duplicates
  [3] "hourlyCalories_merged.csv"                - 0 duplicates
  [4] "hourlyIntensities_merged.csv"             - 0 duplicates
  [5] "hourlySteps_merged.csv"                   - 0 duplicates
  [6] "minuteCaloriesNarrow_merged.csv"          - 0 duplicates
  [7] "minuteIntensitiesNarrow_merged.csv"       - 0 duplicates
  [8] "minuteMETsNarrow_merged.csv"              - 0 duplicates
  [9] "minuteSleep_merged.csv"                   - 525 duplicates
  [10] "minuteStepsNarrow_merged.csv"            - 0 duplicates
  [11] "weightLogInfo_merged.csv"                - 0 duplicates
   names(data_folder_2)
  [1] "dailyActivity_merged.csv"                 - 0 duplicates
  [2] "dailyCalories_merged.csv"                 - 0 duplicates
  [3] "dailyIntensities_merged.csv"              - 0 duplicates
  [4] "dailySteps_merged.csv"                    - 0 duplicates
  [5] "heartrate_seconds_merged.csv"             - 0 duplicates
  [6] "hourlyCalories_merged.csv"                - 0 duplicates
  [7] "hourlyIntensities_merged.csv"             - 0 duplicates
  [8] "hourlySteps_merged.csv"                   - 0 duplicates
```

[9] "minuteCaloriesNarrow_merged.csv"	- 0 duplicates
[10] "minuteCaloriesWide_merged.csv"	- 0 duplicates
[11] "minuteIntensitiesNarrow_merged.csv"	- 0 duplicates
[12] "minuteIntensitiesWide_merged.csv"	- 0 duplicates
[13] "minuteMETsNarrow_merged.csv"	- 0 duplicates
[14] "minuteSleep_merged.csv"	- 543 duplicates
[15] "minuteStepsNarrow_merged.csv"	- 0 duplicates
[16] "minuteStepsWide_merged.csv"	- 0 duplicates
[17] "sleepDay_merged.csv"	- 3 duplicates
[18] "weightLogInfo_merged.csv"	- 0 duplicates

I found 525 duplicates inside *"minuteSleep_merged.csv"* file in folder 1, 543 duplicates inside *"minuteSleep_merged.csv"* file in folder 2 and 3 duplicates inside *"sleepDay_merged.csv"* file in folder 2. I removed them using the following code:

```
df <- distinct(df)  # Remove duplicate rows
```

Step 7: Check & Fix Data Types

To ensure that columns like dates and numbers are stored correctly:

1. **Check column types**

```
str(df)
```

2. **Convert date columns**

If a date column is stored as text, will convert:

```
df$ActivityDate <- as.Date(df$ActivityDate, format="%m/%d/%Y")  # Convert to Date format
```

3. **Convert numeric columns**

If a numeric column is incorrectly stored as text, will fix:

```
df$TotalSteps <- as.numeric(df$TotalSteps)  # Convert to numeric
```

Step 8: Check for Outliers

To find extreme values, i used a boxplot.

```
boxplot(df$TotalSteps, main="Boxplot of Steps")
```

If outliers are extreme, remove them:

```
df <- df %>% filter(TotalSteps < quantile(TotalSteps, 0.99))  # Remove top 1% extreme values
```

Step 9: Save the Cleaned Data

Now that the data is clean, i will save it for analysis.

```
write_csv(df, "C:/Users/Carmen/Desktop/R/cleaned_data_for_analysis.csv")
```

Documentation of Data Cleaning and Manipulation

Source Data

- **Data Location:** CSV files stored on RStudio.
- **Source Files:** Total of 29 CSV files divided into two folders (Folder 1: 11 files, Folder 2: 18 files).

Cleaning and Manipulation Steps

1. **Data Loading:**

- Imported all CSV files from each folder using a custom function in R.

2. **Initial Data Inspection:**

- Reviewed the structure and contents of each dataset using *str()* and *summary().*

- Documented initial data inconsistencies and necessary transformations.

3. **Missing Values Handling:**

- Identified missing values using a custom function:
  - Detected 31 missing values in *“weightLogInfo_merged.csv”* (Folder 1).
  - Detected 65 missing values in *"weightLogInfo_merged.csv"* (Folder 2).
- Removed missing values using *na.omit()* for accurate analysis.

4. **Duplicate Removal:**

- Checked each dataset for duplicates:
  - Found 525 duplicates in *„minuteSleep_merged.csv”* in folder 1, 543 duplicates in *„minuteSleep_merged.csv”* in folder 2 and 3 duplicates in *"sleepDay_merged.csv"* in folder 2.
- Duplicates removed using *distinct()*.

5. **Data Type Correction:**

- Verified column types with *str().*
- Converted incorrect data types:
  - Transformed *ActivityDate* from character to Date type using *as.Date().*
  - Converted numeric columns improperly stored as characters (e.g., *TotalSteps*) to numeric using *as.numeric().*

6. **Outlier Detection and Handling:**

- Visually inspected for outliers using *boxplots* (e.g., *TotalSteps*).
- Removed extreme outliers (top 1%) using *quantile filtering*.

7. **Validation:**

- Conducted final validation by:
  - Reconfirming absence of missing values and duplicates.
  - Ensuring data type consistency.
  - Reviewing statistical distributions and visual summaries.

Resulting Data

- **Clean Dataset Location:** *C:/Users/Carmen/Desktop/R/cleaned_data_for_analysis.csv*
- **Key Columns:** *Id*, *ActivityDate*, *TotalSteps*, *Calories*, *SleepDay*, *TotalMinutesAsleep*
- **Derived Columns:** Adjusted date formats, numeric conversions, cleaned and outlier-filtered numeric variables.

4. ANALYZE

STEP 1: **OPENING RSTUDIO AND SETTING UP ENVIRONMENT**

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

Then loading the libraries:

```
library(tidyverse)
library(lubridate)
```

STEP 2: IMPORTING MY DATA

```
df <- read_csv("cleaned_data_for_analysis.csv")
```

Verify clearly:

```
head(df)
```

```
str(df)
```

STEP 3: FORMATTING AND ADDING DAY OF WEEK

Clearly convert date format:

```
df <- df %>%
  mutate(ActivityDate = as.Date(ActivityDate))
```

Add weekday clearly:

```
df <- df %>%
  mutate(DayOfWeek = wday(ActivityDate, label = TRUE))
```

Check clearly:

```
head(df$DayOfWeek)
```

STEP 4: PERFORM WEEKLY AGGREGATION

Aggregate average values by weekday clearly:

```
weekly_summary <- df %>%
  group_by(DayOfWeek) %>%
  summarize(
    AvgSteps = mean(TotalSteps, na.rm = TRUE),
    AvgCalories = mean(Calories, na.rm = TRUE),
    AvgSedentaryMinutes = mean(SedentaryMinutes, na.rm = TRUE),
    AvgVeryActiveMinutes = mean(VeryActiveMinutes, na.rm = TRUE)
  )
```

Check clearly:

```
print(weekly_summary)
```

STEP 5: CORRELATION ANALYSIS

Clearly calculate and display correlations:

```
cor_steps_calories <- cor(df$TotalSteps, df$Calories, use = "complete.obs")
cor_steps_sedentary <- cor(df$TotalSteps, df$SedentaryMinutes, use = "complete.obs")
print(paste("Correlation (Steps vs Calories):", round(cor_steps_calories, 2)))
print(paste("Correlation (Steps vs Sedentary Minutes):", round(cor_steps_sedentary, 2)))
```

STEP 6: VISUALIZATIONS

Create simple visualizations clearly:

Average Steps by Day of Week

```
ggplot(weekly_summary, aes(x=DayOfWeek, y=AvgSteps)) +
  geom_col(fill="lightblue") +
  labs(title="Average Steps by Day of Week", x="Day of Week", y="Average Steps")
```

Correlation (Steps vs Calories)

```
ggplot(df, aes(x=TotalSteps, y=Calories)) +
  geom_point(alpha=0.5, color="darkblue") +
  geom_smooth(method='lm') +
  labs(title="Correlation between Steps and Calories Burned", x="Total Steps", y="Calories")
```

Correlation (Steps vs Sedentary minutes)

```
ggplot(df, aes(x = TotalSteps, y = SedentaryMinutes)) +
  geom_point(alpha = 0.5, color = "darkred") +
  geom_smooth(method = 'lm', color = "blue") +
  labs(title = "Correlation between Steps and Sedentary Minutes",
       x = "Total Steps",
       y = "Sedentary Minutes") +
  theme_minimal()
```

The resulting dataset from the analysis has the following structure:
Total entries: 452 rows
Columns available clearly:

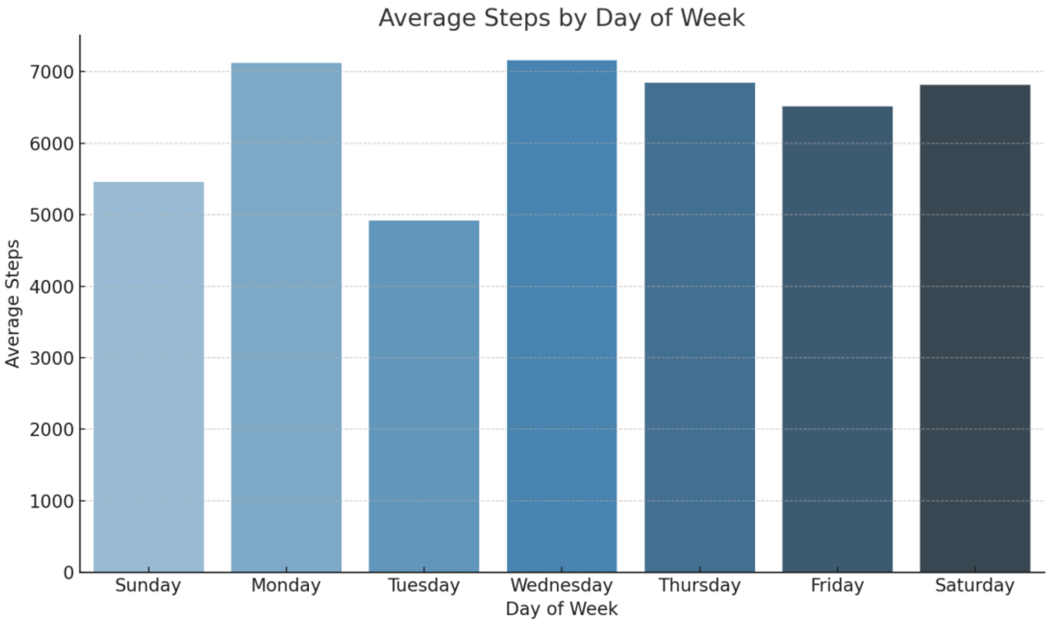
- Id (user identifier)
- ActivityDate (date)
- Activity metrics: TotalSteps, TotalDistance, TrackerDistance, etc.
- Active minutes metrics: VeryActiveMinutes, FairlyActiveMinutes, LightlyActiveMinutes, SedentaryMinutes
- Calories burned (Calories)

I've clearly summarized the data in a table called *"weekly_activity_summary.csv"*, imported from Rstudio, displaying average daily steps, calories, sedentary minutes, and very active minutes per weekday.

SPECIFIC INSIGHTS FROM ANALYSIS:

1. Weekly Patterns:

Highest Average Steps:
Wednesday (7,157 steps) and Monday (7,119 steps) are the most active days.
Lowest Average Steps:
Tuesday (4,915 steps) and Sunday (5,457 steps) have notably lower activity levels.

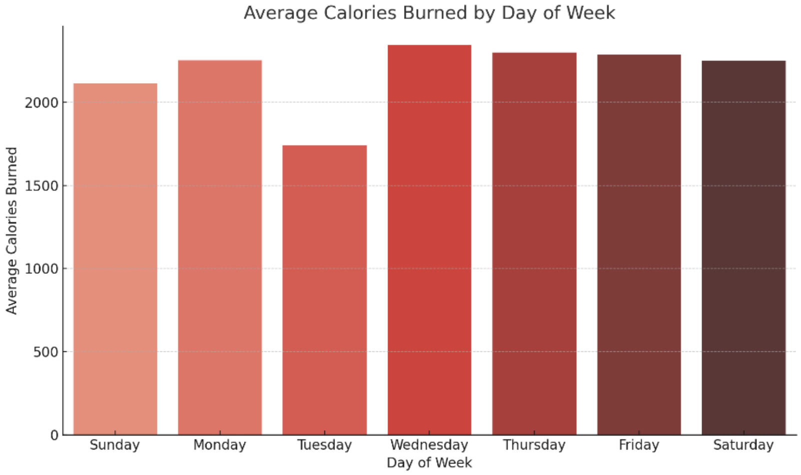


Users show higher activity levels early-to-midweek (Monday and Wednesday), with notable decreases during weekends and especially on Sundays.

Calories burned directly correlate with daily step counts, meaning more active days naturally lead to higher calorie expenditure.

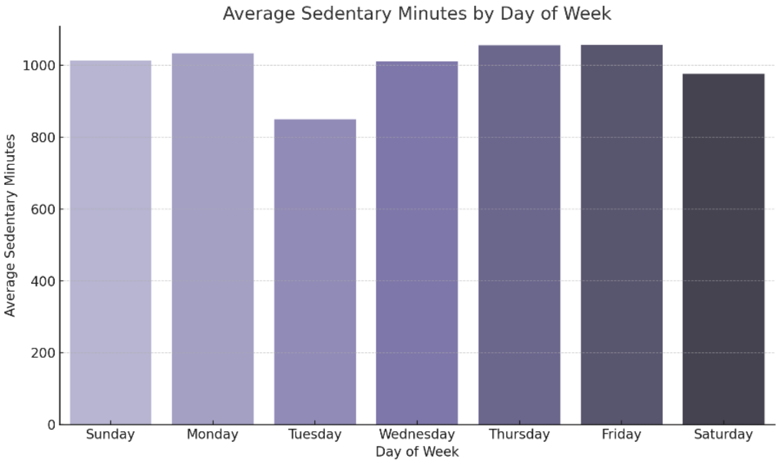
2. Calories Burned:

Highest on Wednesday (average 2,342 calories), aligning with higher activity levels.



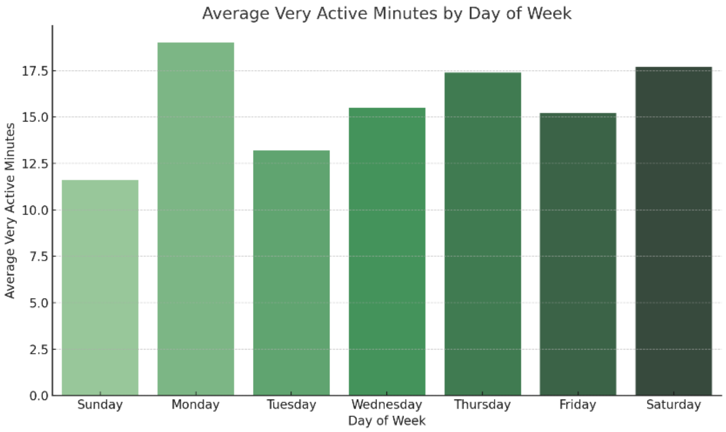
3. Sedentary Behavior:

Increased sedentary minutes are common towards the end of the week (Thursday – 1,055 min and Friday – 1,056 min), possibly reflecting fatigue or busier work schedules.



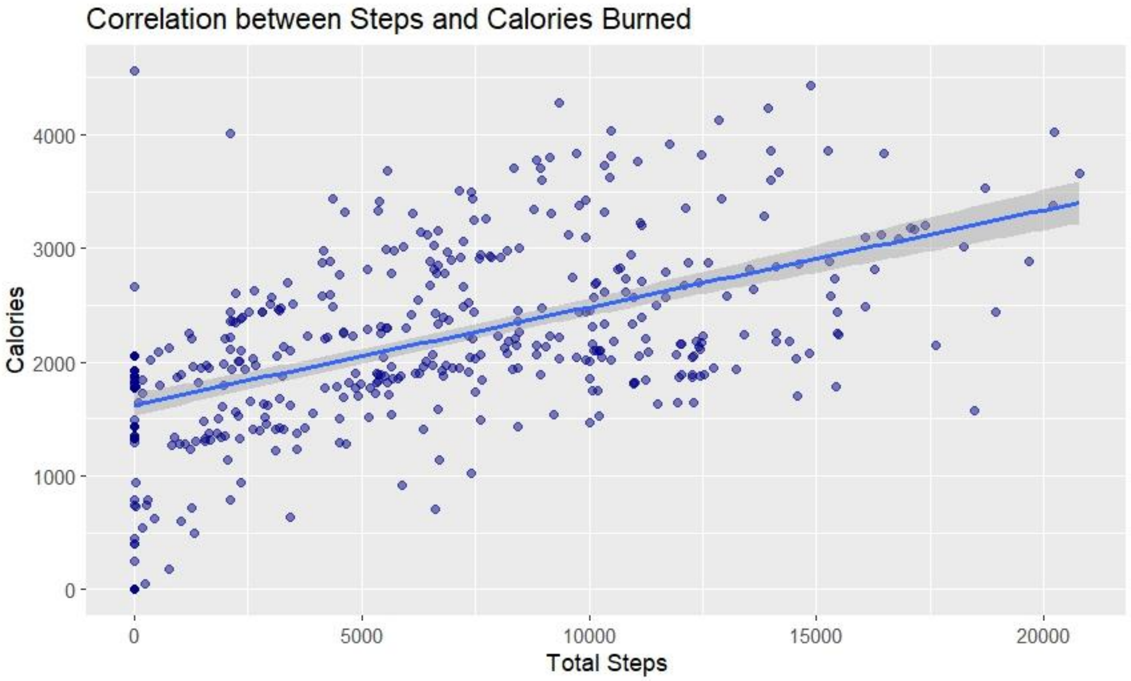
4. Very Active Minutes:

Highest very active minutes occur on Monday (19 min) and Saturday (18 min), suggesting these might be preferred days for workouts or higher-intensity activities.

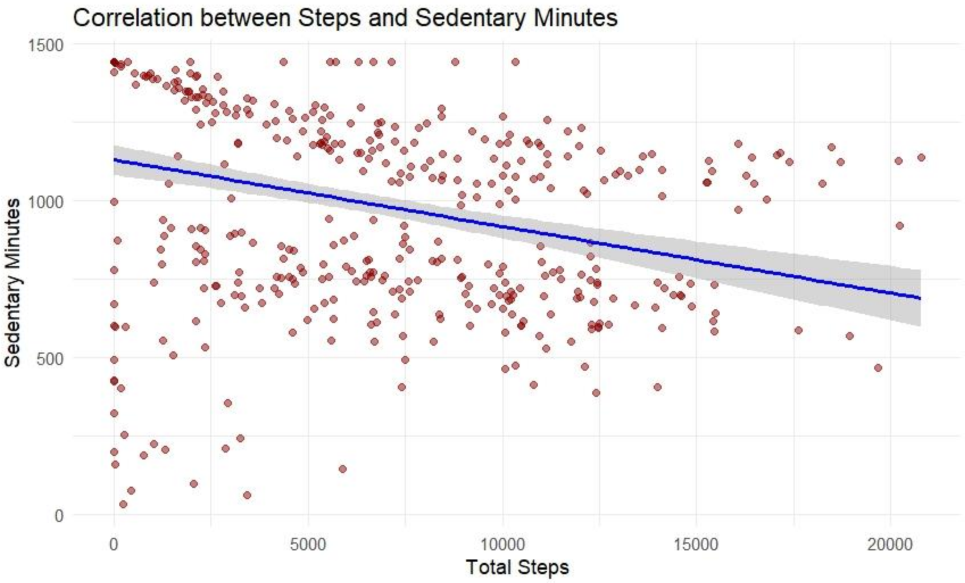


CORRELATION ANALYSIS:

Steps vs Calories Burned: Moderate positive correlation (0.54), indicating more steps typically lead to higher calorie expenditure.



Steps vs Sedentary Minutes: Moderate negative correlation (-0.32), suggesting increased physical activity naturally reduces sedentary time.



5. SHARE

Dragos_Andrei_SPATAREANU_Bellabeat_Case_Study_Presentation.pptx
Weekly_Activity_Summary.csv
Average_Calories_Burned_By_Day_Of_Week.png
Average_Sedentary_Minutes_By_Day_Of_Week.png
Average_Steps_By_Day_Of_Week.png
Average_Very_Active_Minutes_By_Day_Of_Week.png
Correlation_Between_Steps_And_Calories_Burned.jpeg
Correlation_Between_Steps_And_Sedentary_Minutes.jpeg

6. ACT

SUMMARY OF INSIGHTS:

Weekly Activity Patterns: Users show peak activities early-to-midweek (Monday, Wednesday) and reduced activity levels during weekends, especially Sunday.

Calories and Activity Correlation: A clear relationship exists between daily step counts and calories burned, suggesting activity encouragement positively impacts wellness.

Sedentary Behavior: Sedentary behavior increases toward the end of the week (Thursday, Friday), possibly reflecting work fatigue or lifestyle patterns.

Active Minutes: Intentional high-intensity activities typically occur early in the week (Monday) and mid-weekend (Saturday).

RECOMMENDATIONS FOR BELLABEAT:

1. Personalized user recommendations:

Given that average daily steps decrease by approximately 30% on Sundays (5,457 steps) compared to Wednesdays (7,157 steps), Bellabeat should send targeted activity reminders specifically on Sundays.

2. Product integration strategies:

Bellabeat Leaf:

Considering the moderate negative correlation (-0.32) between steps and sedentary minutes, integrating hourly movement reminders into the Leaf product can help users reduce sedentary behavior by prompting short, frequent walks

3. Marketing and user experience optimization:

With a moderate positive correlation (0.54) between daily steps and calories burned, marketing campaigns should highlight how small increases in daily steps can significantly boost calorie expenditure.

Schedule social media campaigns, promotional emails, and product launches strategically around identified peak engagement days (Monday, Wednesday) to maximize user interaction and brand visibility.

SELECTED PRODUCT FOR IMPLEMENTATION: BELLABEAT LEAF

Ideal for applying these insights, it provides stylish yet effective activity, sleep, and wellness tracking. Positioning the Leaf to specifically address sedentary behavior aligns closely with users' lifestyle needs identified through analysis.