How can a Wellness Technology Company Play It Smart?

Summary

Bellabeat, founded in 2013 by Urška Sršen and Sando Mur, is a tech-driven wellness company focused on health-related smart products for women. Sršen's artistic background contributed to Bellabeat's distinctive, aesthetically pleasing designs, which are crafted to empower women by providing insights into their activity, sleep, stress, and reproductive health. Rapid growth led to global offices by 2016 and multiple product launches, with Bellabeat products sold on their website and various online retailers. The company employs a broad marketing strategy, with a strong emphasis on digital marketing through platforms like Google, Facebook, Instagram, Twitter, and YouTube, alongside traditional media channels.

Sršen has tasked the marketing analytics team with analyzing data from Bellabeat devices to better understand current user behaviors. The goal is to use these insights to develop high-level marketing recommendations that align with observed trends, aiming to further optimize Bellabeat's strategy for continued growth.

Ask Phase

Business Task

Sršen asks you to analyze smart device usage data in order to gain insight into how consumers use non-Bellabeat smart devices. She then wants you to select one Bellabeat product to apply these insights to in your presentation. These questions
will guide your analysis:

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?

Stakeholder

- Urška Sršen - Bellabeat cofounder and Chief Creative Officer
- Sando Mur - Bellabeat cofounder and key member of Bellabeat executive team
- Bellabeat Marketing Analytics team

Prepare Phase

Dataset used

Our case study uses the Fitbit Fitness Tracker Data, which is hosted on Kaggle and provided by Mobius

Accessibility and privacy of data

Upon verifying the dataset's metadata, we confirmed it is open-source. The owner has released it into the public domain, waiving all copyright and related rights globally to the fullest extent permitted by law. This allows unrestricted copying, modification, distribution, and use, including for commercial purposes, without needing permission.

Information about our dataset

These datasets were created from responses to a distributed survey conducted via Amazon Mechanical Turk between 12/03/2016 and 12/05/2016. Thirty eligible Fitbit users consented to submit their personal tracker data, which includes minute-level data on physical activity, heart rate, and sleep. Variations in the data reflect differences in Fitbit tracker types and individual tracking behaviors/preferences.

Data Organization and verification

We have access to 18 CSV files, each containing different types of quantitative data tracked by Fitbit. The dataset is considered long-format, with each row representing a single time point for each user, meaning each user has multiple rows based on tracked days and times. Every user is identified by a unique ID across these entries.

Given the small sample size, I organized and filtered the data using Pivot Tables in Excel. This allowed me to verify the attributes and observations within each file, examine relationships between tables, count the sample size (number of users) in each dataset, and confirm the analysis period of 31 days.

Table Name	Type	Description
dailyActivity_merged	Microsoft Excel CSV	Daily Activity over 31 days of 33 users. Tracking daily: Steps, Distance, Intensities, Calories
dailyCalories_merged	Microsoft Excel CSV	Daily Calories over 31 days of 33 users
dailyIntensities_merged	Microsoft Excel CSV	Daily Intensity over 31 days of 33 users. Measured in Minutes and Distance, dividing groups in 4 categories: Sedentary, Lightly Active, Fairly Active, Very Active
dailySteps_merged	Microsoft Excel CSV	Daily Steps over 31 days of 33 users
heartrate_seconds_merged	Microsoft Excel CSV	Exact day and time heartrate logs for just 7 users
hourlyCalories_merged	Microsoft Excel CSV	Hourly Calories burned over 31 days of 33 users
hourlyIntensities_merged	Microsoft Excel CSV	Hourly total and average intensity over 31 days of 33 users
hourlySteps_merged	Microsoft Excel CSV	Hourly Steps over 31 days of 33 users
minuteCaloriesNarrow_merged	Microsoft Excel CSV	Calories burned every minute over 31 days of 33 users (Every minute in single row)
minuteCaloriesWide_merged	Microsoft Excel CSV	Calories burned every minute over 31 days of 33 users (Every minute in single column)
minuteIntensitiesNarrow_merged	Microsoft Excel CSV	Intensity counted by minute over 31 days of 33 users (Every minute in single row)
minuteIntensitiesWide_merged	Microsoft Excel CSV	Intensity counted by minute over 31 days of 33 users (Every minute in single column)
minuteMETsNarrow_merged	Microsoft Excel CSV	Ratio of the energy you are using in a physical activity compared to the energy you would use at rest. Counted in minutes
minuteSleep_merged	Microsoft Excel CSV	Log Sleep by Minute for 24 users over 31 days. Value column not specified

minuteStepsNarrow_merged	Microsoft Excel CSV	Steps tracked every minute over 31 days of 33 users (Every minute in single row)
minuteStepsWide_merged	Microsoft Excel CSV	Steps tracked every minute over 31 days of 33 users (Every minute in single column)
sleepDay_merged	Microsoft Excel CSV	Daily sleep logs, tracked by: Total count of sleeps a day, Total minutes, Total Time in Bed
weightLogInfo_merged	Microsoft Excel CSV	Weight track by day in Kg and Pounds over 30 days. Calculation of BMI.5 users report weight manually 3 users not. In total there are 8 users

Data Credibility and Integrity

Given the dataset's small sample size (30 users) and lack of demographic information, there is potential for sampling bias, making it unclear whether the sample represents the broader population. Additionally, the dataset is somewhat outdated, and the survey's limited duration (only two months) presents further challenges for generalization.

Process Phase

I will conduct my analysis in R because of its accessibility, the volume of data, and its capability to create data visualizations for effectively sharing results with stakeholders.

Installing packages and opening libraries

We will choose the packages that will help us on our analysis and open them. We will use the following packages for our analysis:

```
library(ggpubr)
library(tidyverse)
library(here)
library(skimr)
library(janitor)
library(lubridate)
library(ggrepel)
```

Importing datasets

Based on the available datasets, we will upload those most relevant to answering our business task. Our analysis will focus on the following datasets:

- Daily_activity
- Daily_sleep
- Hourly_steps

Given the small sample sizes, we will exclude the Weight (8 users) and Heart Rate (7 users) datasets from this analysis.

```
daily_activity ← read.csv("/kaggle/input/fitbit/mturkfitbit_export_4.12.16-5.12.16/Fitabase Data 4.12.16-5.12.16/dailyActivi
daily_sleep ← read.csv("/kaggle/input/fitbit/mturkfitbit_export_4.12.16-5.12.16/Fitabase Data 4.12.16-5.12.16/sleepDay_m
hourly_steps ← read.csv("/kaggle/input/fitbit/mturkfitbit_export_4.12.16-5.12.16/Fitabase Data 4.12.16-5.12.16/hourlyStep
```

Preview our datasets

We will preview our selected data frames and check the summary of each column.

```
head(daily_activity)
str(daily_activity)

head(daily_sleep)
str(daily_sleep)
```

```
head(hourly_steps)
str(hourly_steps)
```

Cleaning and formatting

Having gained a better understanding of our data structures, we will now process them to identify any errors and inconsistencies.

Verifying number of users

Before proceeding with our data cleaning, we want to determine the number of unique users in each data frame. Although 30 is the minimum sample size, we will retain the sleep dataset for practice purposes only.

```
n_unique(daily_activity$Id)
n_unique(daily_sleep$Id)
n_unique(hourly_steps$Id)
```

33, 24, 33

Duplicates

We will now look for any duplicates:

```
sum(duplicated(daily_activity))
sum(duplicated(daily_sleep))
sum(duplicated(hourly_steps))
```

0, 3, 0

Remove duplicates and N/A

```
daily_activity ← daily_activity %>%
  distinct() %>%
  drop_na()

daily_sleep ← daily_sleep %>%
  distinct() %>%
  drop_na()

hourly_steps ← hourly_steps %>%
  distinct() %>%
  drop_na()
```

We will verify that duplicates have been removed

```
sum(duplicated(daily_sleep))
```

0

Clean and rename columns

We want to ensure that the column names have the correct syntax and consistent formatting across all datasets, as we will be merging them later. We will convert all column names to lowercase.

```
clean_names(daily_activity)
daily_activity← rename_with(daily_activity, tolower)
clean_names(daily_sleep)
daily_sleep ← rename_with(daily_sleep, tolower)
```

```
clean_names(hourly_steps)
hourly_steps ← rename_with(hourly_steps, tolower)
```

Consistency of date and time columns

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

daily_sleep ← daily_sleep %>%
   rename(date = sleepday)
   mutate(date = as_date(date, format="%m/%d/%Y %I:%M:%S %p"))
```

We will check our cleaned datasets

```
head(daily_activity)
head(daily_sleep)
```

For our hourly_steps dataset, we will convert the date string to a date-time format

```
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()))
head(hourly_steps)
```

Merging Datasets

We will merge the daily_activity and daily_sleep datasets to examine any correlations between variables, using the ID and date as primary keys.

```
daily_activity_sleep ← merge(daily_activity, daily_sleep, by=c ("id", "date"))
glimpse(daily_activity_sleep)
```

Analyze Phase and Share Phase

We will analyze trends among Fitbit users to assess how this information can inform BellaBeat's marketing strategy.

Type of users per activity level

User Classification by Activity Level As our sample lacks demographic variables, we aim to identify user types based on the available data. We can categorize users according to their daily step counts. Most pedometers follow the 10,000-step guideline, but it is also possible to adjust your routine to achieve 7,500 steps daily. The classification of activity levels is as follows:

- Sedentary: Fewer than 5,000 steps per day
- Low active: Approximately 5,000 to 7,499 steps per day
- Somewhat active: Approximately 7,500 to 9,999 steps per day
- Active: More than 10,000 steps per day
- Highly active: More than 12,500 steps per day

This classification is based on information from the following article:

https://www.medicinenet.com/how_many_steps_a_day_is_considered_active/article.htm

First we will calculate the daily steps average by user

```
daily_average ← daily_activity_sleep %>%
  group_by(id) %>%
```

```
  summarise (mean_daily_steps = mean(totalsteps), mean_daily_calories = mean(calories), mean_daily_sleep = mean(t

head(daily_average)
```

id	mean_daily_steps	mean_daily_calories	mean_daily_sleep
<dbl>	<dbl>	<dbl>	<dbl>
1503960366	12405.680	1872.280	360.2800
1644430081	7967.750	2977.750	294.0000
1844505072	3477.000	1676.333	652.0000
1927972279	1490.000	2316.200	417.0000
2026352035	5618.679	1540.786	506.1786
2320127002	5079.000	1804.000	61.0000

We will now classify our users by the daily average steps

```
user_type ← daily_average %>%
   mutate(user_type = case_when(mean_daily_steps < 5000 ~ "sedentary",
   mean_daily_steps >= 5000 & mean_daily_steps < 7499 ~ "lightly active",
   mean_daily_steps >= 7500 & mean_daily_steps < 9999 ~ "fairly active",
   mean_daily_steps >= 10000 ~ "very active", mean_daily_steps >= 12500 ~ "Highly active"))
head(user_type)
```

id	mean_daily_steps	mean_daily_calories	mean_daily_sleep	user_type
<dbl>	<dbl>	<dbl>	<dbl>	<chr>
1503960366	12405.680	1872.280	360.2800	very active
1644430081	7967.750	2977.750	294.0000	fairly active
1844505072	3477.000	1676.333	652.0000	sedentary
1927972279	1490.000	2316.200	417.0000	sedentary
2026352035	5618.679	1540.786	506.1786	lightly active
2320127002	5079.000	1804.000	61.0000	lightly active

Now that we have a new column with the user type we will create a data frame with the percentage of each user type to better visualize them on a graph.

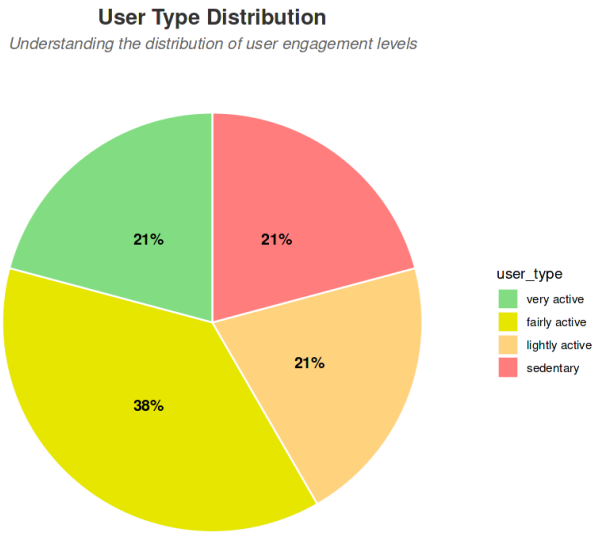
```
# Calculate the percentage of each user type
user_type_percent ← user_type %>%

  # Group the data by user type
  group_by(user_type) %>%

  # Summarize to get the total number of users in each type
  summarise(total = n()) %>%

  # Create a new column that holds the total number of users across all types
  mutate(totals = sum(total)) %>%

  # Group by user type again (though it's already grouped)
  group_by(user_type) %>%

  # Calculate the percentage of each user type by dividing the count by the total
  summarise(total_percent = total / totals) %>%

  # Create a new column with formatted percentage labels
  mutate(labels = scales::percent(total_percent))
```

```
# Convert user_type column to a factor with specified levels for ordered categories
user_type_percent$user_type ← factor(user_type_percent$user_type, levels = c("very active", "fairly active", "lightly a

# Display the first few rows of the resulting data frame
head(user_type_percent)
```

user_type	total_percent	labels
<fct>	<dbl>	<chr>
fairly active	0.3750000	38%
lightly active	0.2083333	21%
sedentary	0.2083333	21%
very active	0.2083333	21%

Below we can see that users are fairly distributed by their activity considering the daily amount of steps. **We can determine that based on users activity all kind of users wear smart-devices**.

```
library(ggplot2)

user_type_percent %>%
  ggplot(aes(x = "", y = total_percent, fill = user_type)) +

  # Create a bar chart and convert it to a pie chart
  geom_bar(stat = "identity", width = 1, color = "white") +  # Add white border to bars for separation
  coord_polar("y", start = 0) +

  # Minimal theme for a clean look
  theme_minimal() +

  # Customize theme elements for aesthetics
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    panel.border = element_blank(),
    panel.grid = element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_blank(),
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold", color = "#333333"),  # Centered title
    plot.subtitle = element_text(hjust = 0.5, size = 12, face = "italic", color = "#666666")  # Centered and styled subtitle
  ) +

  # Custom color palette
  scale_fill_manual(values = c("#85e085", "#e6e600", "#ffd480", "#ff8080")) +

  # Add labels with better positioning and aesthetics
  geom_text(aes(label = labels),
            position = position_stack(vjust = 0.5),
            color = "black",  # Change text color to white for contrast
            size = 4,         # Adjust size for readability
            fontface = "bold") +  # Make labels bold for better visibility
  labs(
    title = "User Type Distribution",
    subtitle = "Understanding the distribution of user engagement levels"  # Subtitle will be centered
  )
```



Steps and minutes asleep per weekday

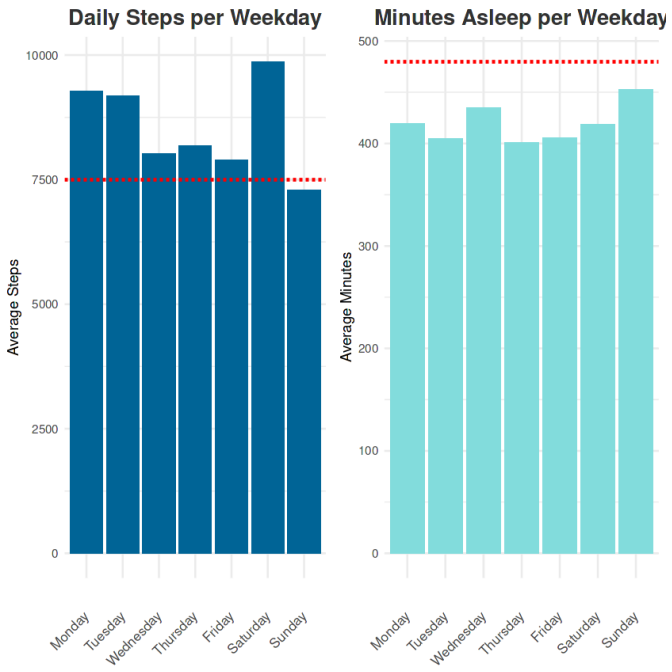
We aim to identify which days of the week users are most active and when they tend to sleep the most. Additionally, we will check if users are meeting the recommended daily step count and sleep duration.

Below, we are calculating the weekdays based on our date column, as well as determining the average number of steps taken and minutes slept for each weekday.

```
#  Steps and minutes asleep per weekday
# Create a new data frame with weekdays derived from the date
weekday_steps_sleep ← daily_activity_sleep %>%
  mutate(weekday = weekdays(date))  # Extract the weekday names from the date column

# Convert the weekday column into an ordered factor with specific levels
weekday_steps_sleep$weekday ← ordered(weekday_steps_sleep$weekday, levels = c("Monday", "Tuesday", "Wedne

# Group the data by weekday and calculate mean steps and sleep for each weekday
weekday_steps_sleep ← weekday_steps_sleep %>%
  group_by(weekday) %>%  # Group data by the weekday column
  summarize(daily_steps = mean(totalsteps),  # Calculate average total steps per weekday
            daily_sleep = mean(totalminutesasleep))  # Calculate average total sleep per weekday

# Display the first few rows of the resulting data frame
head(weekday_steps_sleep)
```

weekday	daily_steps	daily_sleep
<ord>	<dbl>	<dbl>
Monday	9273.217	419.5000
Tuesday	9182.692	404.5385
Wednesday	8022.864	434.6818
Thursday	8183.516	401.2969
Friday	7901.404	405.4211
Saturday	9871.123	419.0702

```
library(ggplot2)
library(ggpubr)

# Arrange the two plots in a wider layout with equal widths
ggarrange(
  # Plot for daily steps
  ggplot(weekday_steps_sleep) +
    geom_col(aes(weekday, daily_steps), fill = "#006699") +  # Dark blue fill for bars
    geom_hline(yintercept = 7500, linetype = "dashed", color = "red", size = 1) +  # Dashed red line for goal
    labs(title = "Daily Steps per Weekday",
         x = "",
         y = "Average Steps") +  # Clear y-axis label
    theme_minimal() +  # Use a minimal theme for a clean look
    theme(
      axis.text.x = element_text(angle = 45, vjust = 0.5, hjust = 1, size = 10),  # Adjust x-axis text
      plot.title = element_text(hjust = 0.5, size = 16, face = "bold", color = "#333333")  # Centered title
    ),

  # Plot for minutes asleep
  ggplot(weekday_steps_sleep, aes(weekday, daily_sleep)) +
    geom_col(fill = "#85e0e0") +  # Light blue fill for bars
    geom_hline(yintercept = 480, linetype = "dashed", color = "red", size = 1) +  # Dashed red line for goal
    labs(title = "Minutes Asleep per Weekday",
         x = "",
         y = "Average Minutes") +  # Clear y-axis label
    theme_minimal() +  # Use a minimal theme for a clean look
    theme(
      axis.text.x = element_text(angle = 45, vjust = 0.5, hjust = 1, size = 10),  # Adjust x-axis text
      plot.title = element_text(hjust = 0.5, size = 16, face = "bold", color = "#333333")  # Centered title
    ),

  # Adjust the width of the plots
  ncol = 2,  # Number of columns in the arrangement
  widths = c(1, 1)  # Set equal widths for both plots
)
```



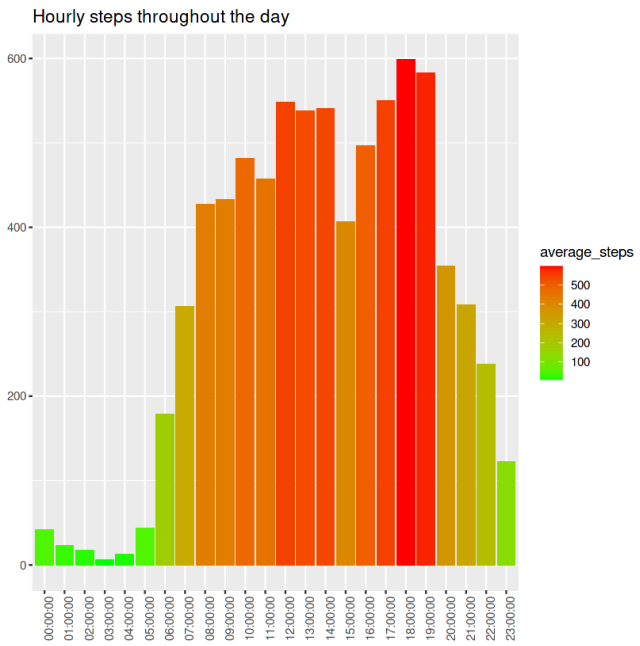
The graphs above show that:

- Users walk the recommended daily amount of 7,500 steps, except on Sundays.
- Users do not get the recommended 8 hours of sleep each night.

Hourly steps throughout the day

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

head(hourly_steps)
```

```
# Group the hourly_steps dataset by the 'time' column
hourly_steps %>%
  group_by(time) %>%
  # Calculate the average steps for each time group
  summarize(average_steps = mean(steptotal)) %>%
  # Create a ggplot object
  ggplot() +
  # Add a column plot for average steps, mapping 'time' to the x-axis and 'average_steps' to the y-axis
  geom_col(mapping = aes(x = time, y = average_steps, fill = average_steps)) +
  # Add labels for the plot title and axes
  labs(title = "Hourly Steps Throughout the Day", x = "", y = "") +
  # Create a gradient fill for the bars, transitioning from green (low steps) to red (high steps)
  scale_fill_gradient(low = "green", high = "red") +
  # Adjust the x-axis text to be vertical for better readability
  theme(axis.text.x = element_text(angle = 90))
```



The data indicates that users are most active between 8 AM and 7 PM, with a noticeable increase in step count during lunchtime from 12 PM to 2 PM and again in the evenings from 5 PM to 7 PM.

Next, we will analyze the correlation between the following pairs of variables:

- Daily steps and daily sleep
- Daily steps and calories

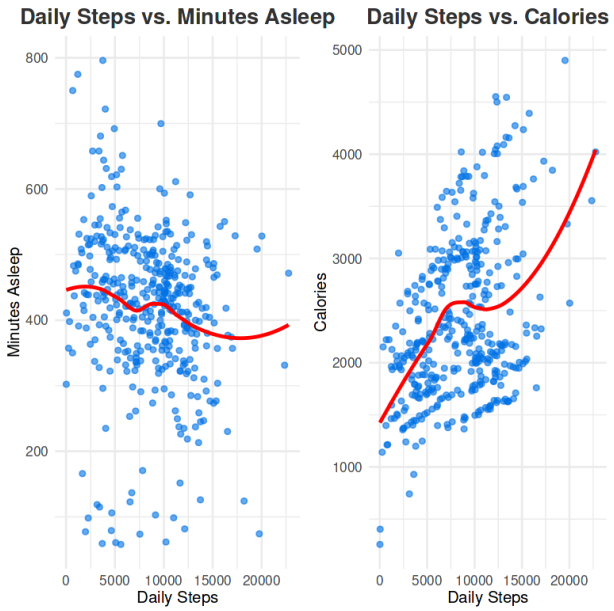
Correlations

We will now determine if there is any correlation between different variables:

- Daily steps and daily sleep
- Daily steps and calories

```
library(ggplot2)
library(ggpubr)

# Create the first plot: Daily Steps vs. Minutes Asleep
steps_vs_sleep_plot ← ggplot(daily_activity_sleep, aes(x = totalsteps, y = totalminutesasleep)) +
  geom_jitter(color = "#0073e6", alpha = 0.6, width = 0.2) +  # Jittered points with color and transparency
  geom_smooth(color = "red", se = FALSE, linetype = "solid") +  # Smooth line without confidence interval
  labs(title = "Daily Steps vs. Minutes Asleep",
       x = "Daily Steps",
       y = "Minutes Asleep") +
  theme_minimal() +  # Use a minimal theme for a clean look
  theme(
    panel.background = element_blank(),  # Clear panel background
    plot.title = element_text(size = 16, face = "bold", color = "#333333", hjust = 0.5),  # Centered title
    axis.title = element_text(size = 12),  # Consistent axis title size
    axis.text = element_text(size = 10)  # Consistent axis text size
  )

# Create the second plot: Daily Steps vs. Calories
steps_vs_calories_plot ← ggplot(daily_activity_sleep, aes(x = totalsteps, y = calories)) +
  geom_jitter(color = "#0073e6", alpha = 0.6, width = 0.2) +  # Jittered points with color and transparency
  geom_smooth(color = "red", se = FALSE, linetype = "solid") +  # Smooth line without confidence interval
```

```
  labs(title = "Daily Steps vs. Calories",
       x = "Daily Steps",
       y = "Calories") +
  theme_minimal() +  # Use a minimal theme for a clean look
  theme(
    panel.background = element_blank(),  # Clear panel background
    plot.title = element_text(size = 16, face = "bold", color = "#333333", hjust = 0.5),  # Centered title
    axis.title = element_text(size = 12),  # Consistent axis title size
    axis.text = element_text(size = 10)  # Consistent axis text size
  )

# Arrange the two plots in a wider layout with larger dimensions
ggarrange(steps_vs_sleep_plot, steps_vs_calories_plot,
          ncol = 2,  # Number of columns
          nrow = 1,  # Number of rows
          widths = c(1.2, 1.2),  # Adjust the width of each plot
          heights = c(1)) +  # Adjust the height of the layout
  theme(plot.margin = margin(20, 20, 20, 20))  # Add margin around the plots
```



According to our plots:

- There is no correlation between daily activity levels, as measured by steps taken, and the number of minutes users sleep each day.
- However, there is a positive correlation between steps taken and calories burned. As expected, an increase in the number of steps is associated with a higher number of calories burned.

Use of smart device

Days used smart device

Now that we have observed trends in activity, sleep, and calories burned, we want to examine how frequently users in our sample utilize their devices. This information will help us develop our marketing strategy and identify features that could enhance the use of smart devices.

We will calculate the number of users who use their smart devices daily, categorizing our sample into three groups based on a 31-day period:

- High Use: Users who utilize their device between 21 and 31 days.
- Moderate Use: Users who utilize their device between 10 and 20 days.
- Low Use: Users who utilize their device between 1 and 10 days.

First, we will create a new data frame that groups by ID, calculates the number of days each device was used, and adds a new column for the classification described above.

```
daily_use ← daily_activity_sleep %>%
  group_by(id) %>%
  summarize(days_used=sum(n())) %>%
  mutate(usage = case_when(
    days_used >= 1 & days_used <= 10 ~ "low use",
    days_used >= 11 & days_used <= 20 ~ "moderate use",
    days_used >= 21 & days_used <= 31 ~ "high use",
  ))

head(daily_use)
```

id	days_used	usage
<dbl>	<int>	<chr>
1503960366	25	high use
1644430081	4	low use
1844505072	3	low use
1927972279	5	low use
2026352035	28	high use
2320127002	1	low use

We will now create a percentage data frame to better visualize the results in the graph. We are also ordering our usage levels.

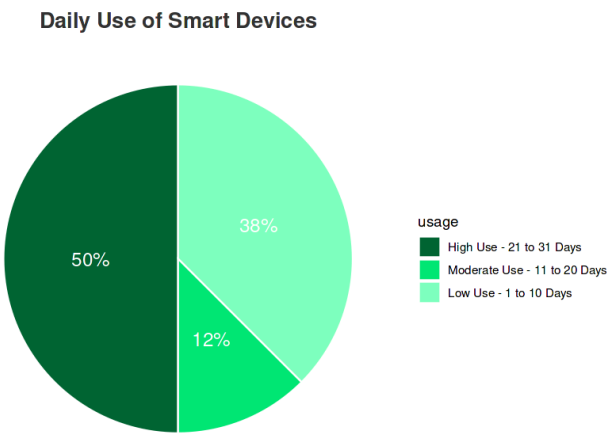
```
daily_use_percent ← daily_use %>%
  group_by(usage) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(usage) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))

daily_use_percent$usage ← factor(daily_use_percent$usage, levels = c("high use", "moderate use", "low use"))

head(daily_use_percent)
```

usage	total_percent	labels
<fct>	<dbl>	<chr>
high use	0.500	50%
low use	0.375	38%
moderate use	0.125	12%

Now that we have our new table we can create our plot:

```
library(ggplot2)

# Create a pie chart for daily smart device usage with a right-aligned caption
```

```
daily_use_percent %>%
  ggplot(aes(x = "", y = total_percent, fill = usage)) +
  geom_bar(stat = "identity", width = 1, color = "white") +  # Add a white border to the bars
  coord_polar("y", start = 0) +
  theme_minimal() +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    panel.border = element_blank(),
    panel.grid = element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_blank(),
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold", color = "#333333"),
    plot.caption = element_text(hjust = 1, size = 10, color = "#555555", vjust = -1)  # Right-aligned caption
  ) +
  geom_text(aes(label = labels), position = position_stack(vjust = 0.5), color = "#FFFFFF", size = 5) +  # Text color for c
  scale_fill_manual(values = c("#006633", "#00e673", "#80ffbf"),
                    labels = c("High Use - 21 to 31 Days",
                               "Moderate Use - 11 to 20 Days",
                               "Low Use - 1 to 10 Days")) +
  labs(title = "Daily Use of Smart Devices")
```



Analyzing our results we can see that

- 50% of the users of our sample use their device frequently - between 21 to 31 days.
- 12% use their device 11 to 20 days.
- 38% of our sample use really rarely their device

Time used smart device

To be more specific, we aim to analyze the number of minutes users wear their devices each day. To achieve this, we will merge the daily_use data frame with daily_activity allowing us to filter the results based on the daily usage of the device.

```
daily_use_merged ← merge(daily_activity, daily_use, by=c ("id"))
```

```
head(daily_use_merged)
```

id	date	totalsteps	totaldistance	trackerdistance	loggedactivitiesdistance	veryactivedistance
	<dbl>	<date>	<dbl>	<dbl>	<dbl>	<dbl>
1	1503960366	2016-05-07	11992	7.71	7.71	0
2	1503960366	2016-05-06	12159	8.03	8.03	0
3	1503960366	2016-05-01	10602	6.81	6.81	0
4	1503960366	2016-04-30	14673	9.25	9.25	0
5	1503960366	2016-04-12	13162	8.50	8.50	0
6	1503960366	2016-04-13	10735	6.97	6.97	0

We need to generate a new data frame that calculates the total number of minutes users wore their devices each day and categorizes them into three groups:

- **All Day:** The device was worn for the entire day.
- **More than Half Day:** The device was worn for more than half of the day.
- **Less than Half Day:** The device was worn for less than half of the day.

```
minutes_worn ← daily_use_merged %>%
  mutate(total_minutes_worn = veryactiveminutes+fairlyactiveminutes+lightlyactiveminutes+sedentaryminutes)%>%
  mutate (percent_minutes_worn = (total_minutes_worn/1440)*100) %>%
  mutate (worn = case_when(
    percent_minutes_worn == 100 ~ "All day",
    percent_minutes_worn < 100 & percent_minutes_worn >= 50~ "More than half day",
    percent_minutes_worn < 50 & percent_minutes_worn > 0 ~ "Less than half day"
  ))

head(minutes_worn)
```

id	date	totalsteps	totaldistance	trackerdistance	loggedactivitiesdistance	veryactivedistance
	<dbl>	<date>	<dbl>	<dbl>	<dbl>	<dbl>
1	1503960366	2016-05-07	11992	7.71	7.71	0
2	1503960366	2016-05-06	12159	8.03	8.03	0
3	1503960366	2016-05-01	10602	6.81	6.81	0
4	1503960366	2016-04-30	14673	9.25	9.25	0
5	1503960366	2016-04-12	13162	8.50	8.50	0
6	1503960366	2016-04-13	10735	6.97	6.97	0

As we have done before, to better visualize our results we will create new data frames. In this case we will create four different data frames to arrange them later on on a same visualization.

- First data frame will show the total of users and will calculate percentage of minutes worn the device taking into consideration the three categories created.
- The three other data frames are filtered by category of daily users so that we can see also the difference of daily use and time use.

```
minutes_worn_percent← minutes_worn%>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(worn) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))
```

```
minutes_worn_highuse ← minutes_worn%>%
  filter (usage == "high use")%>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(worn) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))

minutes_worn_moduse ← minutes_worn%>%
  filter(usage == "moderate use") %>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(worn) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))

minutes_worn_lowuse ← minutes_worn%>%
  filter (usage == "low use") %>%
  group_by(worn) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(worn) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))

minutes_worn_highuse$worn ← factor(minutes_worn_highuse$worn, levels = c("All day", "More than half day", "Less
minutes_worn_percent$worn ← factor(minutes_worn_percent$worn, levels = c("All day", "More than half day", "Less t
minutes_worn_moduse$worn ← factor(minutes_worn_moduse$worn, levels = c("All day", "More than half day", "Less
minutes_worn_lowuse$worn ← factor(minutes_worn_lowuse$worn, levels = c("All day", "More than half day", "Less th

head(minutes_worn_percent)
head(minutes_worn_highuse)
head(minutes_worn_moduse)
head(minutes_worn_lowuse)
```

worn	total_percent	labels
<fct>	<dbl>	<chr>
All day	0.36465638	36%
Less than half day	0.03506311	4%
More than half day	0.60028050	60%

worn	total_percent	labels
<fct>	<dbl>	<chr>
All day	0.06756757	6.8%
Less than half day	0.04324324	4.3%
More than half day	0.88918919	88.9%

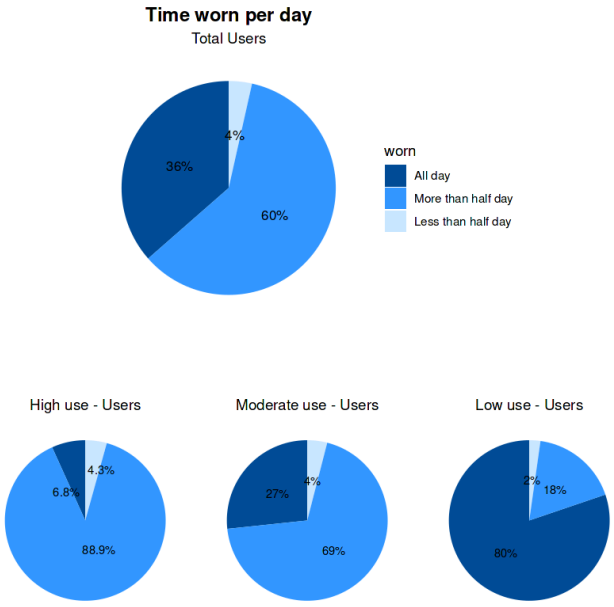
worn	total_percent	labels
<fct>	<dbl>	<chr>
All day	0.2666667	27%
Less than half day	0.0400000	4%
More than half day	0.6933333	69%

worn	total_percent	labels
<fct>	<dbl>	<chr>
All day	0.80223881	80%
Less than half day	0.02238806	2%
More than half day	0.17537313	18%

Now that we have created the four data frames and also ordered worn level categories, we can visualize our results in the following plots. All the plots have been arranged together for a better visualization.

```
ggarrange(
  ggplot(minutes_worn_percent, aes(x="",y=total_percent, fill=worn)) +
  geom_bar(stat = "identity", width = 1)+
  coord_polar("y", start=0)+
  theme_minimal()+
  theme(axis.title.x= element_blank(),
        axis.title.y = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        plot.title = element_text(hjust = 0.5, size=14, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5)) +
    scale_fill_manual(values = c("#004d99", "#3399ff", "#cce6ff"))+
  geom_text(aes(label = labels),
            position = position_stack(vjust = 0.5), size = 3.5)+
  labs(title="Time worn per day", subtitle = "Total Users"),
  ggarrange(
  ggplot(minutes_worn_highuse, aes(x="",y=total_percent, fill=worn)) +
  geom_bar(stat = "identity", width = 1)+
  coord_polar("y", start=0)+
  theme_minimal()+
  theme(axis.title.x= element_blank(),
        axis.title.y = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        plot.title = element_text(hjust = 0.5, size=14, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5),
        legend.position = "none")+
    scale_fill_manual(values = c("#004d99", "#3399ff", "#cce6ff"))+
  geom_text_repel(aes(label = labels),
            position = position_stack(vjust = 0.5), size = 3)+
  labs(title="", subtitle = "High use - Users"),
  ggplot(minutes_worn_moduse, aes(x="",y=total_percent, fill=worn)) +
  geom_bar(stat = "identity", width = 1)+
  coord_polar("y", start=0)+
  theme_minimal()+
  theme(axis.title.x= element_blank(),
        axis.title.y = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        plot.title = element_text(hjust = 0.5, size=14, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5),
        legend.position = "none") +
```

```
  scale_fill_manual(values = c("#004d99", "#3399ff", "#cce6ff"))+
geom_text(aes(label = labels),
          position = position_stack(vjust = 0.5), size = 3)+
labs(title="", subtitle = "Moderate use - Users"),
ggplot(minutes_worn_lowuse, aes(x="",y=total_percent, fill=worn)) +
geom_bar(stat = "identity", width = 1)+
coord_polar("y", start=0)+
theme_minimal()+
theme(axis.title.x= element_blank(),
      axis.title.y = element_blank(),
      panel.border = element_blank(),
      panel.grid = element_blank(),
      axis.ticks = element_blank(),
      axis.text.x = element_blank(),
      plot.title = element_text(hjust = 0.5, size=14, face = "bold"),
      plot.subtitle = element_text(hjust = 0.5),
      legend.position = "none") +
  scale_fill_manual(values = c("#004d99", "#3399ff", "#cce6ff"))+
geom_text(aes(label = labels),
          position = position_stack(vjust = 0.5), size = 3)+
labs(title="", subtitle = "Low use - Users"),
ncol = 3),
nrow = 2)
```



Based on our analysis, we observe that 36% of users wear their devices all day, while 60% wear them for more than half the day, and only 4% wear them for less than half the day.

When we filter the total number of users by the number of days they have used the device, we can see how long they wear it each day, yielding the following insights:

Usage Categories:

High Use: Users who wear their device for 21 to 31 days.

Moderate Use: Users who wear their device for 10 to 20 days.

Low Use: Users who wear their device for 1 to 10 days. Among high users, only 6.8% wear their device all day, while a substantial 88.9% wear it for more than half the day but not the entire day.

In contrast, moderate users tend to wear their devices less frequently on a daily basis. Interestingly, low users tend to wear their devices for longer periods on the days they do use them.

Conclusion

Based on the analysis of smart device usage trends, here are some key insights and recommendations for Bellabeat:

1. Trends in smart device usage

- 50% of users are high-frequency users, wearing their devices for 21-31 days
- 60% of users wear their devices for more than half the day
- Users are most active between 8 AM and 7 PM, with peaks during lunchtime (12 PM - 2 PM) and evenings (5 PM - 7 PM)
- There's a positive correlation between steps taken and calories burned
- Most users achieve the recommended 7,500 daily steps, except on Sundays
- Users generally don't get the recommended 8 hours of sleep per night

2. Application to Bellabeat customers

- Bellabeat can focus on designing comfortable devices that can be worn for extended periods
- Emphasize features that track activity during peak hours (8 AM - 7 PM)
- Incorporate calorie tracking alongside step counting
- Develop sleep tracking features to help users improve their sleep habits

3. Influencing Bellabeat's marketing strategy

- Highlight the comfort and long-wearing capability of Bellabeat devices
- Promote features that support active lifestyles, especially during typical work hours
- Emphasize the connection between activity and calorie burn in marketing materials
- Create campaigns around improving sleep habits and the importance of sleep tracking
- Develop strategies to encourage device use on less active days, like Sundays
- Target marketing efforts towards health-conscious individuals who are interested in tracking their daily activity and improving their overall well-being

By focusing on these trends and tailoring their products and marketing strategies accordingly, Bellabeat can position itself as a company that truly understands and caters to the needs of health-conscious consumers.