Bellabeat smart device usage case study

In this case study, my goal is to analyze smart device usage data to provide Bellabeat with insights that can help them improve their products and better understand their users' wellness habits. Using FitBit Fitness Tracker Data, the aim is to focus on key health metrics like activity levels and sleep. My analytical approach follows the Data Analytics Process, utilizing the steps of **Ask, Prepare, Process, Analyze, Share, and Act**. As a result of this case study, I'll offer data-driven recommendations to support Bellabeat's mission of empowering healthier lifestyles.

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**, co-founder 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.

Products

- **Bellabeat app:** The Bellabeat app provides users with health data related to their activity, sleep, stress, menstrual cycle, and mindfulness habits. This data can help users better understand their current habits and make healthy decisions. The Bellabeat app connects to their line of smart wellness products
- **Leaf:** Bellabeat's classic wellness tracker can be worn as a bracelet, necklace, or clip. The Leaf tracker connects to the Bellabeat app to track activity, sleep, and stress
- **Time:** This wellness watch combines the timeless look of a classic timepiece with smart technology to track user activity, sleep, and stress. The Time watch connects to the Bellabeat app to provide you with insights into your daily wellness.
- **Spring:** This is a water bottle that tracks daily water intake using smart technology to ensure that you are appropriately hydrated throughout the day. The Spring bottle connects to the Bellabeat app to track your hydration levels.
- **Bellabeat membership:** Bellabeat also offers a subscription-based

membership program for users. Membership gives users 24/7 access to fully personalized guidance on nutrition, activity, sleep, health and beauty, and mindfulness based on their lifestyle and goals.

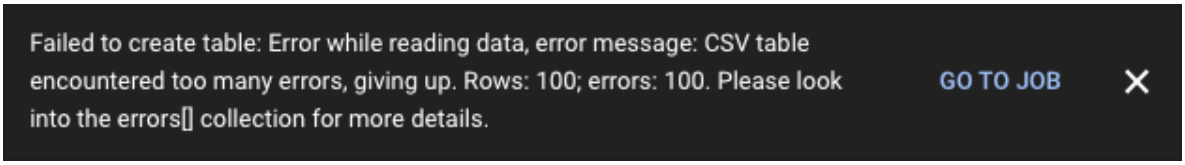
Ask

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

- What are some trends in smart device usage?
- How could these trends apply to Bellabeat customers?
- How could these trends help influence Bellabeat marketing strategy?

Prepare

I will be using FitBit Fitness Tracker Data, pointed by Sršen. This dataset contains personal fitness tracker data from thirty Fitbit users consented to the submission of personal tracker data, including minute-level output for physical activity, heart rate, and sleep monitoring. It includes information about daily activity, steps, and heart rate that can be used to explore users' habits.
Even though this dataset is outdated and has limitations, since it is a case study, I'm not going to point out it's incredibility.
This dataset consists 29 csv files, so initial idea was to upload everything to **BigQuery** and start working on it with **SQL.** Unfortunately, trying to upload first file already gave me an error:



So, It was time to move on to heavy tools - **Python**

Process

While processing data, I used **Python Pandas** library. After inspecting some files from current dataset, I noticed that date/time columns have formatted in a way which is not recognized by BiqQuery. In addition, these columns have various different names throughout dataset ("ActivityHour", "date", "Date" etc.)

```
import pandas as pd

file_path = '/Users/mac...'

df = pd.read_csv(file_path)
data_types = df.dtypes

print(df.head())
print(data_types)
```

```
           Id            ActivityHour  Calories
0  1503960366  4/12/2016 12:00:00 AM          81
1  1503960366   4/12/2016 1:00:00 AM          61
2  1503960366   4/12/2016 2:00:00 AM          59
3  1503960366   4/12/2016 3:00:00 AM          47
4  1503960366   4/12/2016 4:00:00 AM          48
Id                int64
ActivityHour     object
Calories          int64
dtype: object
```

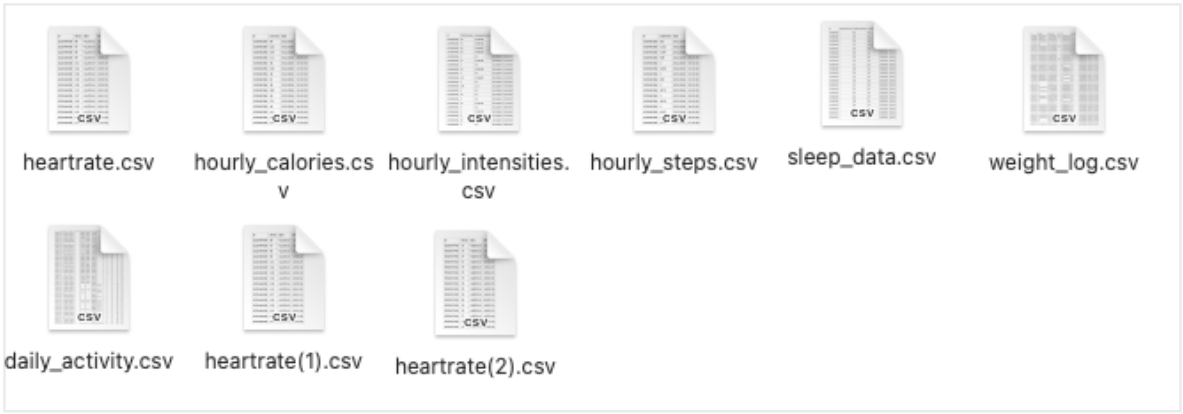
So I wrote a script to create new csv files with date in “date” column and time in “time” column:

```python
import pandas as pd

file_name = 'hourly_intensities.csv'
file_path = f'/Users/mac...{file_name}'
column_name = 'ActivityHour'
new_path = '/Users/mac...'

df = pd.read_csv(file_path)

# Convert 'ActivityHour' column to pandas datetime
df[column_name] = pd.to_datetime(df[column_name])

# Create two new columns for date and time
df['date'] = df[column_name].dt.strftime('%-m/%-d/%-Y')
df['time'] = df[column_name].dt.strftime('%H:%M:%S')

# Delete existing 'ActivityHour' column
df.drop(columns=[column_name], inplace=True)

# Create new csv file
df.to_csv(f'/Users/mac...{file_name}', index=False)

print(f'Created new file: {file_name}')
```

I created separate date and time columns specifically because I want to use SQL for further analysis. This makes it easier to reference the necessary values without having to extract them each time, in my opinion.
Since this dataset contains data about two different monthly periods, it would be wise to merge them into one:

```python
import pandas as pd

# Display all existing columns in dataframe. For example when using head() function.
pd.set_option('display.max_columns', None)

# Paths to csv files
march_april_data = '/Users/mac...'
april_may_data = '/Users/mac...'

# Creating separate dataframes from these files
df = pd.read_csv(march_april_data)
df2 = pd.read_csv(april_may_data)

# Just in case, check how many unique Id's there are in both datasets.
unique_ids = df['Id'].nunique()
unique_ids2 = df2["Id"].nunique()

print(df.shape, unique_ids)
print(df2.shape, unique_ids2)

# Merge two datasets
merged_data = pd.merge(df, df2, how='outer')

# Double check
unique_ids3 = merged_data['Id'].nunique()
print(merged_data.shape, unique_ids3)
print(merged_data.tail())

# Now let's create a new CSV file from our merged data
output_path = '/Users/mac...'
merged_data.to_csv(output_path, index=False)

print(f'\nData saved to {output_path}')
```

After fixing datetime format and merging the files, I had nine files to start working with, so I started uploading them into BigQuery:



One of the files, specifically ‘heartrate.csv’ had more than 3.5million rows, and therefore it’s filesize was greater than 100MB. This made it unable to upload into BigQuery (I’m using BigQuery Sandbox) so I had to be creative. I split the file into two, using Python:

```python
import pandas as pd


df = pd.read_csv('/Users/mac...')

split_index = 1800000

df1 = df.iloc[:split_index]
df2 = df.iloc[split_index:]

df1.to_csv('/Users/mac...', index=False)
df2.to_csv('/Users/mac...', index=False)

print('CSV has been split into two parts')
```

This made me able to create two separate tables in BigQuery as ‘heartrate1’ and ‘heartrate2’, which I later merged into one ‘heartrate’ table again.

Great! Now I had my data in BigQuery and I was ready to dig into it by using SQL.
As I was exploring ‘daily_activity table’ , I noticed that there are multiple rows with all zero values, so:

```sql
SELECT
  *
FROM
  `testingdaata.fitbit_data.daily_activity_full`
WHERE
  TotalSteps = 0
```

This resulted in 138 rows of zero values. As shown in the image below, most of these rows contained some numeric value in ‘SedentaryMinutes’ column. Many of these values were 1440, which is 24 hours. If value was less, I assume that battery ran out. Either way, this meant that the device was not used on that day, so I deleted these rows.

Row	date	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDist	VeryActiveDistance	ModeratelyActiveDis	LightActiveDistance	SedentaryActiveDist	VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedentaryMinutes
25	2016-05-11	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	1440
26	2016-05-12	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	966
27	2016-05-05	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	1440
28	2016-05-08	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	1440
29	2016-05-09	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	1440
30	2016-04-01	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	1440
31	2016-04-02	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	1440
32	2016-04-03	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	1440
33	2016-04-04	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0	0	1440

I also searched for NULL and duplicate values. Although I didn't find any duplicates, there were many NULL values, which based on the context, I decided to remove or not. Since SQL itself doesn't have a direct function to check all columns for NULL in one go, I decided to use Python again. Instead of typing out each column in my query, I used Python 'for' loop:

```
import pandas as pd

file_path = '/Users/mac...'

df = pd.read_csv(file_path)

for name in df.columns:
    print(f'{name} IS NULL OR')
```

```
Id IS NULL OR
date IS NULL OR
TotalSteps IS NULL OR
TotalDistance IS NULL OR
TrackerDistance IS NULL OR
LoggedActivitiesDistance IS NULL OR
VeryActiveDistance IS NULL OR
ModeratelyActiveDistance IS NULL OR
LightActiveDistance IS NULL OR
SedentaryActiveDistance IS NULL OR
VeryActiveMinutes IS NULL OR
FairlyActiveMinutes IS NULL OR
LightlyActiveMinutes IS NULL OR
SedentaryMinutes IS NULL OR
Calories IS NULL OR
```

Easy copy/paste to SQL query:

```sql
SELECT
  *
FROM
  `testingdaata.fitbit_data.daily_activity`
WHERE
  Id IS NULL OR
  date IS NULL OR
  TotalSteps IS NULL OR
  TotalDistance IS NULL OR
  TrackerDistance IS NULL OR
  LoggedActivitiesDistance IS NULL OR
  VeryActiveDistance IS NULL OR
  ModeratelyActiveDistance IS NULL OR
  LightActiveDistance IS NULL OR
  SedentaryActiveDistance IS NULL OR
  VeryActiveMinutes IS NULL OR
  FairlyActiveMinutes IS NULL OR
  LightlyActiveMinutes IS NULL OR
  SedentaryMinutes IS NULL OR
  Calories IS NULL
```

After going through each table using similar methods, I was ready for analyzis.

Analyze

In this phase, I will use SQL to further manipulate and organize my data. I will save my findings as local csv files, which later can be imported into Google Sheets for visualizations.
First of all, health organizations, including the CDC, align with the general guideline of **10,000 steps per day** as a target for maintaining good health. How about our individuals?

```sql
SELECT
  Id,
  COUNT(Id) AS total_count,
  COUNTIF(TotalSteps > 10000) AS count_above_10000,
  ROUND((COUNTIF(TotalSteps > 10000)/COUNT(Id))*100, 2) AS percentage_10000,
  AVG(ROUND((COUNTIF(TotalSteps > 10000)/COUNT(Id))*100, 2)) OVER() AS average_percentage_10000
FROM `testingdaata.fitbit_data.daily_activity`
GROUP BY
  Id
ORDER BY
  percentage_10000 DESC;
```

Row	Id	total_count	count_above_10000	percentage_10000	average_percentage
1	1503960366	49	45	91.84	32.1548484848...
2	8877689391	43	39	90.7	32.1548484848...
3	8053475328	42	37	88.1	32.1548484848...
4	2022484408	43	37	86.05	32.1548484848...
5	7007744171	36	29	80.56	32.1548484848...
6	6962181067	45	34	75.56	32.1548484848...
7	4388161847	31	23	74.19	32.1548484848...
8	3977333714	42	28	66.67	32.1548484848...
9	2347167796	32	20	62.5	32.1548484848...
10	5553957443	43	20	46.51	32.1548484848...
11	7086361926	42	19	45.24	32.1548484848...
12	5577150313	39	13	33.33	32.1548484848...
13	8378563200	43	14	32.56	32.1548484848...
14	6117666160	31	9	29.03	32.1548484848...
15	1644430081	40	11	27.5	32.1548484848...
16	4702921684	44	12	27.27	32.1548484848...
17	4319703577	42	8	19.05	32.1548484848...
18	8253242879	24	4	16.67	32.1548484848...
19	8583815059	38	6	15.79	32.1548484848...
20	6775888955	25	3	12.0	32.1548484848...
21	4020332650	49	5	10.2	32.1548484848...
22	4558609924	43	4	9.3	32.1548484848...
23	2026352035	43	3	6.98	32.1548484848...
24	1624580081	50	3	6.0	32.1548484848...
25	2320127002	39	2	5.13	32.1548484848...
26	2873212765	42	1	2.38	32.1548484848...
27	4445114986	46	0	0.0	32.1548484848...
28	1844505072	30	0	0.0	32.1548484848...
29	1927972279	29	0	0.0	32.1548484848...
30	4057192912	22	0	0.0	32.1548484848...
31	8792009665	31	0	0.0	32.1548484848...
32	3372868164	30	0	0.0	32.1548484848...
33	6290855005	26	0	0.0	32.1548484848...

Out of our 33 participants, only **32.15%** are able to maintain an average of 10,000 daily steps. This is not exactly an ideal result.
Let’s inspect further. I would like to know, when exactly our participants are actively making steps. While inspecting ‘hourly_steps’ table, I noticed that it needs some more cleaning to do. I found 250 days recorded by different Id’s, where 0 steps where made throughout the day:

```sql
SELECT
  Id,
  date
FROM
  `testingdaata.fitbit_data.hourly_steps`
GROUP BY
  Id, date
HAVING
  SUM(StepTotal) = 0
ORDER BY
  date, Id;
```

Row	Id	date
1	1844505072	2016-03-12
2	4057192912	2016-03-12
3	6391747486	2016-03-12
4	7007744171	2016-03-12
5	8253242879	2016-03-12
6	8792009665	2016-03-12
7	1844505072	2016-03-13
8	2320127002	2016-03-13
9	4057192912	2016-03-13

Results per page: 50 1 – 50 of 250

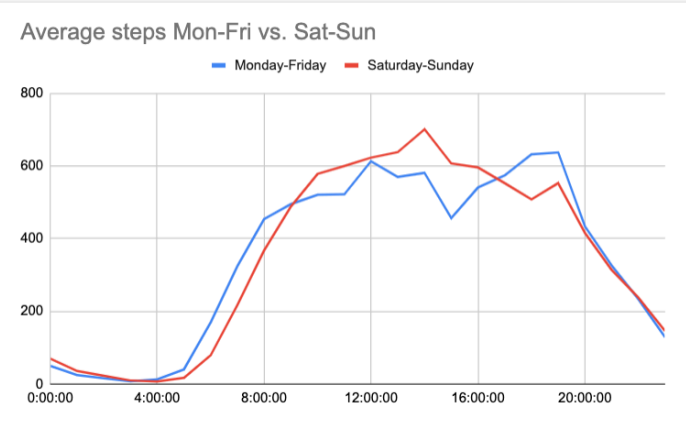
Since I'm using **BigQuery Sandbox** account, I can not manipulate tables by using UPDATE or DROP functions. Instead, I create new table:

```sql
CREATE TABLE `testingdaata.fitbit_data.hourly_steps_2` AS
WITH ZeroSteps AS (
  SELECT
    Id, date
  FROM
    `testingdaata.fitbit_data.hourly_steps`
  GROUP BY
    Id, date
  HAVING
    SUM(StepTotal) = 0
)
SELECT hs.*
FROM `testingdaata.fitbit_data.hourly_steps` hs
LEFT JOIN ZeroSteps zs
ON hs.Id = zs.Id AND hs.date = zs.date
WHERE zs.Id IS NULL;
```

This resulted my table to shrink by 6000 rows of data (250 daily grouped rows * 24 hours = 6000). More importantly, accurate calculations could be performed now. Let's find daily average steps by an hour, keeping weekends/ workdays apart:

```sql
SELECT
  time,
  ROUND(AVG(StepTotal)) as average_steps
FROM `testingdaata.fitbit_data.hourly_steps_2`
WHERE
    FORMAT_DATE('%A', date) != 'Saturday' OR
    FORMAT_DATE('%A', date) != 'Sunday'
GROUP BY
  time
ORDER BY
  time DESC
```

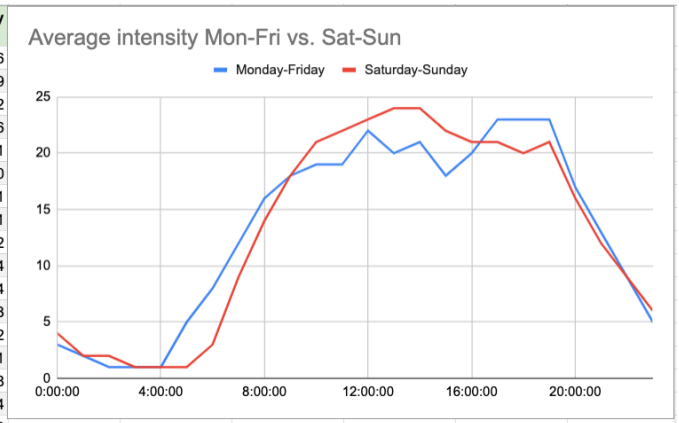
time	average_steps_workday	average_steps_weekend
23:00:00	129	146
22:00:00	234	238
21:00:00	326	314
20:00:00	434	416
19:00:00	637	553
18:00:00	632	508
17:00:00	574	553
16:00:00	541	596
15:00:00	457	607
14:00:00	581	701
13:00:00	570	638
12:00:00	613	623
11:00:00	522	600
10:00:00	521	578
9:00:00	495	488
8:00:00	454	368
7:00:00	324	217
6:00:00	170	79
5:00:00	40	17
4:00:00	13	7
3:00:00	8	10
2:00:00	16	23
1:00:00	25	36
0:00:00	50	70



100% of participants used step-counter feature.

As shown in the graph above, on workdays, participants tended to take the most steps between 6:00 PM and 7:00 PM, which corresponds to the time when most people commute home after work. Another noticeable peak occurred during lunchtime, likely associated with going out for lunch. On weekends, the average steps taken by participants peak at 2 PM, likely due to recreational activities and social gatherings.
Let's compare steps with intensity:

time	average_intensity_workday	average_intensity_weekend
23:00:00	5	6
22:00:00	9	9
21:00:00	13	12
20:00:00	17	16
19:00:00	23	21
18:00:00	23	20
17:00:00	23	21
16:00:00	20	21
15:00:00	18	22
14:00:00	21	24
13:00:00	20	24
12:00:00	22	23
11:00:00	19	22
10:00:00	19	21
9:00:00	18	18
8:00:00	16	14
7:00:00	12	9
6:00:00	8	3
5:00:00	5	1
4:00:00	1	1
3:00:00	1	1
2:00:00	1	2
1:00:00	2	2
0:00:00	3	4



100% of participants used intensity feature.

When comparing the average steps graph with the average intensity graph, the line for workdays remains relatively consistent, indicating that work-related activities were primarily responsible for the intensity during these days. In contrast, the intensity line for weekends is much smoother, suggesting that participants engaged in a variety of activities, including more recreational or sporty pursuits.

Based on the previous analysis, we can conclude that our group of participants does not consistently follow an active lifestyle. To further support this, I ran a

query to calculate the average sedentary hours:

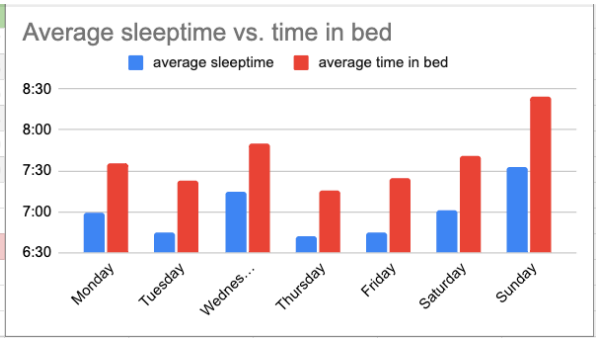
```
SELECT AVG(SedentaryMinutes)/60 FROM `testingdaata.fitbit_data.daily_activity`
```

The result is staggering **15 hours and 51 minutes** of sedentary time per day, far exceeding the recommended range of 6 to 9 hours—and it's important to note that this figure does not include sleep time.

```
SELECT
  FORMAT_DATE('%A', date) AS weekday,
  CONCAT(
    CAST(FLOOR(AVG(TotalMinutesAsleep) / 60) AS STRING), ':',
    LPAD(CAST(MOD(CAST(AVG(TotalMinutesAsleep) AS INT64), 60) AS STRING), 2, '0')
  ) AS average_time_sleep,
  CONCAT(
    CAST(FLOOR(AVG(TotalTimeInBed) / 60) AS STRING), ':',
    LPAD(CAST(MOD(CAST(AVG(TotalTimeInBed) AS INT64), 60) AS STRING), 2, '0')
  ) AS average_time_in_bed
FROM `testingdaata.fitbit_data.sleep`
GROUP BY
  weekday
ORDER BY
  CASE weekday
    WHEN 'Monday' THEN 1
    WHEN 'Tuesday' THEN 2
    WHEN 'Wednesday' THEN 3
    WHEN 'Thursday' THEN 4
    WHEN 'Friday' THEN 5
    WHEN 'Saturday' THEN 6
    WHEN 'Sunday' THEN 7
  END;
```

weekday	average_time_sleep	average_time_in_bed	difference
Monday	6:59	7:36	0:37
Tuesday	6:45	7:23	0:38
Wednesday	7:15	7:50	0:35
Thursday	6:42	7:16	0:34
Friday	6:45	7:25	0:40
Saturday	7:01	7:41	0:40
Sunday	7:33	8:24	0:51

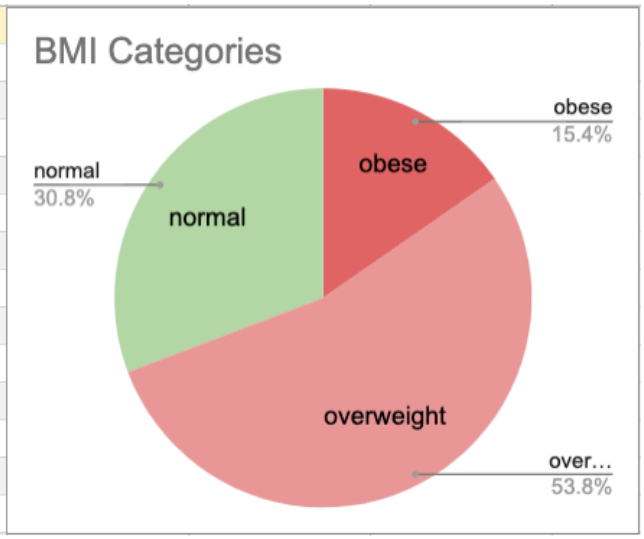
24 distinct **Id**'s which is roughly 78% of all participants.



When reviewing the sleep data, no remarkable findings occure. The average sleep duration is around 7 hours, which is within a healthy range. As expected, participants tend to stay in bed and sleep a bit longer on Sundays.

Finally, I looked at the weight data, which again confirms that our participants are not following healthy lifestyle habits. Even though only 13 participants (around 40%) shared their weight data, the results are not good:

Id	averageBMI	category
1927972279	46.9	obese
4445114986	35	obese
8253242879	29.5	overweight
5577150313	28	overweight
4319703577	27.4	overweight
4558609924	27.2	overweight
4702921684	26.1	overweight
8877689391	25.5	overweight
2891001357	25	overweight
2347167796	24.8	normal
6962181067	24.1	normal
1503960366	22.8	normal
2873212765	21.6	normal



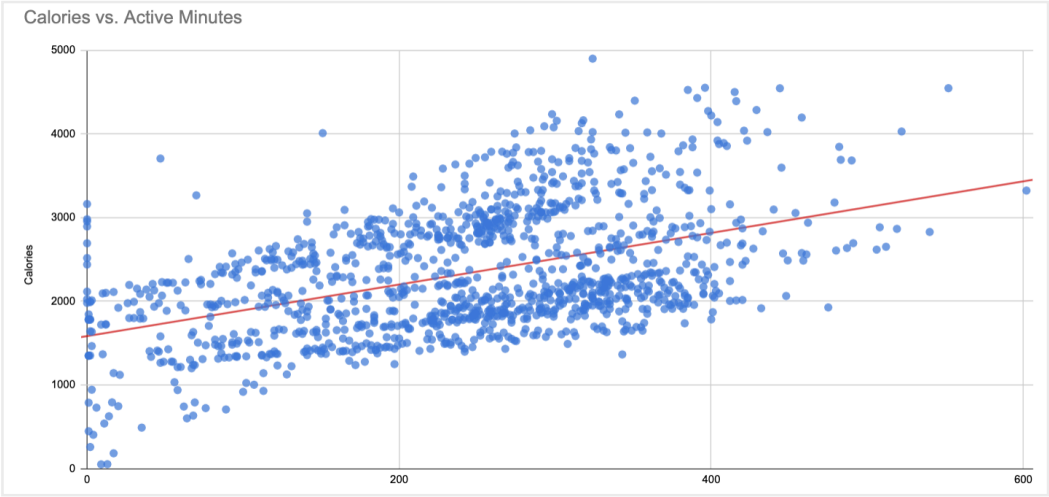
Whopping **69.2%** of our participants are in overweight category. However, it's important to consider that this finding is based on small sample size, which may affect the overall conclusion.

Share & Act

Let's come back to our given tasks:

- What are some trends in smart device usage?
- How could these trends apply to Bellabeat customers?
- How could these trends help influence Bellabeat marketing strategy?

As for trends, we've covered them thoroughly in previous phases, including visualizations. Now, let's draw some general conclusions from the findings. While it would be great to have supporting metrics like age, profession, and demographics, I want to propose a **hypothesis**: most of our participants likely have sedentary office jobs, as their most active times on workdays are during lunchtime and the afternoon commute. A high number of daily sedentary hours further supports this assumption. I'd also take a guess that the average age is 30+, since people in their 20s tend to be more active in the evenings, especially on weekends, a trend we don't see here.
Another trend we can see is overweight among our participants. I created a scatter chart based on 'daily_activity' data table, which clearly shows a positive relationship between active minutes and calories burned. As participants engage in more physical activity, the number of calories burned increases, highlighting the direct impact of active time on energy expenditure and therefore weight loss.



So, we can conclude that Bellabeat should focus on encouraging their customers to be more active in their daily lives. As **Urška Sršen** asked me to explore ways to improve one product in Bellabeat's lineup, I have a few ideas for enhancing the **Bellabeat app**:

- To promote more daily activity, the app could provide personalized notifications based on each user's activity level.
- Every evening at 11:59 PM, the app would generate a performance report for the day, assigning a score to encourage reflection and improvement.
- Milestones, based on each user's activity level, could be set on a daily, weekly, or monthly basis, motivating users to reach specific goals.
- Achievements could be tied to users' personal accounts and shared within the app's community for additional motivation.

On the marketing side, Bellabeat offers a subscription-based **membership** that provides 24/7 access to personalized guidance on nutrition, activity, sleep, health, beauty, and mindfulness based on users' lifestyles and goals.

- To boost customer interest, every new Bellabeat health tracker purchase could come with one month of free membership.
- To retain users, Bellabeat should further invest in a truly personalized experience.
- A leaderboard for top users based on scores and milestones could be introduced, with top performers receiving rewards.

Lastly, I would recommend that Bellabeat gather more detailed data about their customers, which would greatly enhance future analyses. As an analyst, having access to information such as users age, profession, and demographics would provide deeper insights.
Improvements could also be made in the way sleep data is collected. Currently, I only had sleep duration to work with, but additional metrics like sleep onset, wake time, REM sleep, and light sleep would allow for more comprehensive analysis. This, in turn, would enable Bellabeat to offer a more personalized

approach to their users health and wellness.