Google data analytic's capstone project – Bellabeat's casestudy

About the company
Bellabeat is a high-tech company that manufactures health-focused smart products for women. It was founded by Urška Sršen and Sando Mur in 2013. Since it was founded, Bellabeat has grown rapidly and quickly positioned itself as a tech-driven wellness company for women. Collecting data on activity, sleep, stress, and reproductive health has allowed Bellabeat to empower women with knowledge about their own health and habits.

Case scenario
Even though Bellabeat has expanded quickly and established itself as a market leader, the firm is constantly looking for ways to expand and rise to the top of the global digital market. Urška Sršen thinks that by finding out more about the data on smart device usage, Bellabeat may finally be able to fulfill its business objective by discovering new opportunities and insights.

Main objective: The primary objective is to ascertain prospective avenues for expansion and offer suggestions for enhancing Bellabeat's marketing approach by considering patterns in the use of smart devices.

I will follow the data analysis process of **ask**, **prepare**, **process**, **analyze**, **share**, and **act** in order to answer some key business questions.

The tools used: Microsoft Excel, MySQL Workbench and Tableau Public

The remaining of the report describes each of these phases in detail.

Key deliverables: By the end of the analysis, I will produce a report with the following deliverables:

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

Summary of the business task:
The company has a portfolio of five products: bellabeat app, leaf, time, spring and bellabeat membership. Sršen has asked the marketing analytics team to focus on a Bellabeat product and analyze smart device usage data in order to gain insight into how people are already using their smart devices. Then, using this information, she would like high-level recommendations for how these trends can inform Bellabeat marketing strategy.

I am a part of Bellabeat's marketing analytics team. We are a team of data analysts responsible for collecting, analyzing, and repo1ing data that helps guide Bellabeat's marketing strategy. My task as a junior data analyst is to analyze smart device usage data in order to

gain insights into how consumers use non-Bellabeat smart devices. I need to select one Bellabeat product to apply these insights to in my presentation.

Phase 1: ASK

Primary stakeholders: Urška Sršen and Sando Mur, executive team members.

Secondary stakeholders: Bellabeat marketing analytics team.

The questions used to guide the 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 infuence Bellabeat marketing strategy?

Phase 2: Prepare

About the dataset

This Kaggle data set contains personal fitness tracker from thirty unique fitbit users which can be accessed from Kaggle through this link. Thirty eligible Fitbit users agreed to the submission of their personal tracker data, including minute-level output for physical activity, heartrate, and sleep monitoring. It also includes information about daily activity, steps, and heart rate that can be used to explore every users' habits.

The entire dataset is organized in 18 CSV files. The data in each CSV file is stored in the long format. It was collected daily over a period of one month from 12th April to 12th May 2016.

Although there are 18 files, I just used the following five files as it contained all the information required to analyze daily, hourly and sleep related data.

Files chosen:

- dailyActivity_merged.csv
- hourlyCalories_merged.csv
- hourlyActivity_merged.csv
- hourlyIntensities_merged.csv
- sleepDay_merged.csv

The data also follows an ROCCC approach:

- Reliability:Reliability of the data was low. It was collected from 30 FitBit users who gave their consent to the submission of personal tracker data which was generated by from a distributed survey via Amazon Mechanical Turk.
- Original: The originality of the data was low. The data was collected from 30 FitBit users who gave their consent to the submission of personal tracker data.
- Comprehensive: Comprehensiveness of the data was medium. The data was detailed as it included information up to the minute-level output for physical activity, heart

rate, and sleep monitoring. However, the sample size is small and most of the data is recorded during specific days of the week.

- Current: The data was ranked low in terms of how current it was. The data was collected daily over a period of one month from 12th April to 12th May 2016. However, as we are in 2024, therefore the data is outdated as the user's habit might have changed by now. So, if the data is collected now, it might not match with the user's usage behavior in the current time.
- Cited: The data was ranked low in terms of information related to its citation. It was collected from a third party (available by Mobius via Kaggle), so the original citation is unknown.

There are also a few limitations with the dataset.

1. The sample size is very small. It consists of 30 unique users. Usually, a larger sample is preferred for carrying out data analysis.
2. The time period of data collection is very short. Also, the data was collected in 2016, so it is outdated as the usage habits of the 30 users must have changed over a period of time.
3. Three extra users were found which did not record their data for daily activity and sleep.
4. It was found that most of the data was recorded from Tuesday to Thursday, which may not give enough empirical evidence to formulate an accurate analysis.
5. This dataset also lacks demographic information such as age, race, income, etc. This information would prove useful to yield additional perspectives on how diverse user segments engage with the product.

Phase 3: Process

I decided to use MySQL Workbench for processing the data. First, I created a new database called "bellabeat_data" and imported all the relevant files as new tables. I used "Table Data Import Wizard" to import all the data in the tables from their respective files. So, in all, I had 5 tables to work with. The next step was to start cleaning and filtering the data within the tables. The steps are as follows:

1. Cleaning and filtering the data: In this step I removed columns from each table that were irrelevant for my analysis. I found that only dailyactivity_merged table had a few irrelevant columns. The other tables did not have any irrelevant columns, so I used them as it is.

```
-- Remove columns from each table that are irrelevant
use bellabeat_data;
select * from dailyactivity_merged; /*this is the table name */
alter table dailyactivity_merged
drop column LoggedActivitiesDistance,
drop column VeryActiveDistance,
drop column ModeratelyActiveDistance,
drop column LightActiveDistance,
drop column SedentaryActiveDistance;
```

1. Checking the datatype and column names: In this step I checked the datatype and names of all the columns in each table to make sure that every column had the right datatype and names fit the data they contained. I found that the datatype of the column consisting of the date information in each table was 'text' datatype. So, I changed the datatype to date and also renamed that column in each table. While altering the first table, I encountered error code 1175, so I added the safe updates statement in the queries.

```
#Table 1 - dailyactivity_merged
SET SQL_SAFE_UPDATES = 0;
describe dailyactivity_merged;
update dailyactivity_merged
set ActivityDate = str_to_date(ActivityDate, '%m/%d/%Y');
alter table dailyactivity_merged
change column ActivityDate date date;
```

```
#Table 2 - sleepday_merged
SET SQL_SAFE_UPDATES = 0;
describe sleepday_merged;
/* SleepDay column was 'text' datatype, so changed it to 'datetime'
update sleepday_merged
set SleepDay = str_to_date(SleepDay, '%m/%d/%Y %h:%i:%s %p');
alter table sleepday_merged
change column SleepDay date_time datetime;
```

```
#Table 3 - hourlycalories_merged
describe hourlycalories_merged;
SET SQL_SAFE_UPDATES = 0;
/* ActivityHour column was 'text' datatype, so I changed it to 'datetime'
update hourlycalories_merged
set ActivityHour = str_to_date(ActivityHour, '%m/%d/%Y %h:%i:%s %p');
alter table hourlycalories_merged
change column ActivityHour date_time datetime;
```

```
#Table 4 - hourlysteps_merged
SET SQL_SAFE_UPDATES = 0;
describe hourlysteps_merged
/* ActivityHour column was 'text' datatype so I changed it to 'datetime'
/* update hourlysteps_merged
set ActivityHour = str_to_date(ActivityHour, '%Y-%m-%d %H:%i:%s');
alter table hourlysteps_merged
change column ActivityHour date_time datetime; */
```

```sql
#Table 5 - hourlyintensities_merged
describe hourlyintensities_merged;
/* ActivityHour column was 'text' datatype, so I changed it to 'datetime'
update hourlyintensities_merged
set ActivityHour = str_to_date(ActivityHour, '%m/%d/%Y %h:%i:%s %p');
alter table hourlyintensities_merged
change column ActivityHour date_time datetime;
```

2. Checking the total number of distinct users and number of days for each table:

3a. dailyactivity_merged table: In this step I checked the total number of distinct users in the dailyactivity_merged table as well as the total number of days. It came out to be 33 users and 31 days respectively.

```sql
#1 Inspecting 'dailyactivity_merged'
 select * from dailyactivity_merged;
 select count(distinct id) as total_users from dailyactivity_merged;        -- 33 users
 select count(distinct date) as total_days from dailyactivity_merged;        -- 31 days
```

3b. sleepday_merged table: Similar to step 3a, I checked the total number of distinct users in the sleepday_merged table as well and the total number of days. It came out to be 24 users and 31 days respectively.

```sql
#2 Inspecting 'sleepday_merged'
select * from sleepday_merged;
select count(distinct id) as total_users from sleepday_merged;        -- 24 users
select count(distinct date_time) as total_days from sleepday_merged;      -- 31 days
```

3c. hourlycalories_merged table: Similar to steps 3a and 3b, I checked the total number of distinct users in the hourlycalories_merged table as well as and the total number of days. It came out to be 33 users and 31 days respectively.

```sql
#3 Inspecting 'hourlycalories_merged'
-- select * from hourlycalories_merged;
-- select count(*) from hourlycalories_merged;  -- 22099 total records
-- select count(distinct id) as total_users from hourlycalories_merged;       -- 33 users
select count(distinct (date(date_time))) as total_days,
       count(distinct (time(date_time))) as total_hours
from hourlycalories_merged;        -- 31 days (containing 24 hours)
```

3d. hourlyintensities_merged table: Similar to the above steps, I checked the total number of distinct users in the hourlyintensities_merged table as well and the total number of days. It came out to be 33 users and 31 days respectively.

```sql
#4 Inspecting 'hourlyintensities_merged'
-- select * from hourlyintensities_merged;
-- select count(*) from hourlyintensities_merged;  -- 22099 total records
-- select count(distinct id) as total_users from hourlyintensities_merged;    -- 33 users
select count(distinct (date(date_time))) as total_days,
       count(distinct (time(date_time))) as total_hours
from hourlyintensities_merged;        -- 31 days (containing 24 hours)
```

3e. hourlysteps_merged table: Lastly, I checked the total number of distinct users in the hourlysteps_merged table as well and the total number of days. It came out to be 33 users and 31 days respectively.

```
 #5 Inspecting 'hourlysteps_merged'
-- select * from hourlysteps_merged;
-- select count(*) from hourlysteps_merged;  -- 22099 total records
-- select count(distinct id) as total_users from hourlysteps_merged;        -- 33 users
select count(distinct (date(datetime))) as total_days,
count(distinct (time(datetime))) as total_hours from hourlysteps_merged;
-- 31 days (containing 24 hours)
```

3. Removing duplicates and null values: In this step I tried to find duplicate and null values from each table and deleted those records if any, as that data would be of no use for the analysis.

4a. dailyactivity_merged table: In the dailyactivity_merged table, although I found no duplicate values, the total_steps column had 77 observations as '0' which would not be possible because if a person had used the device, he must have walked atleast 1 step that day. Therefore, I removed all the entries with '0' step count. I then crossed checked the accuracy of the data within this table in terms of whether all the activity minutes were adding up to 1440 minutes or not.

```
/* finding duplicates in dailyactivity_merged table*/
select id, date, TotalSteps, TotalDistance, TrackerDistance, Calories, count(*)
from dailyactivity_merged
group by id, date, TotalSteps, TotalDistance, TrackerDistance, Calories
having count(*) > 1;            -- 0 Duplicates as the result was empty
```

```
/* Finding Null/missing values in dailyactivity_merged table*/
SELECT
SUM(CASE WHEN id is null or id = 0 then 1 else 0 end) as missing_id,
    SUM(CASE WHEN date IS NULL THEN 1 ELSE 0 END) AS Missing_ActivityDate,
    SUM(CASE WHEN TotalSteps = 0 THEN 1 ELSE 0 END) AS Zero_TotalSteps,
    SUM(CASE WHEN TotalDistance = 0 THEN 1 ELSE 0 END) AS Zero_TotalDistance,
    SUM(CASE WHEN TrackerDistance = 0 THEN 1 ELSE 0 END) AS Zero_TrackerDistance,
    SUM(CASE WHEN VeryActiveMinutes = 0 THEN 1 ELSE 0 END) AS Zero_VeryActiveMinutes,
    SUM(CASE WHEN FairlyActiveMinutes = 0 THEN 1 ELSE 0 END) AS Zero_FairlyActiveMinutes,
    SUM(CASE WHEN LightlyActiveMinutes = 0 THEN 1 ELSE 0 END) AS Zero_LightlyActiveMinutes,
    SUM(CASE WHEN SedentaryMinutes = 0 THEN 1 ELSE 0 END) AS Zero_SedentaryMinutes,
    SUM(CASE WHEN Calories = 0 THEN 1 ELSE 0 END) AS Zero_Calories
      from dailyactivity_merged;
```

```sql

/* ID and date column as expected didn't have any missing values,
but total_steps column had 77 values as 0 which can't be right
because if a person has used the device he must have walked atleast
1 step that day. So I removed these entries (with 0 step count) */
delete from dailyactivity_merged where TotalSteps = 0;
/* if totalsteps can't be 0 then calories also can't be zero */
select count(*) from dailyactivity_merged
where Calories = 0;       -- no entries with 0 calories
```

```sql
/* Checking if all the activity minutes add upto 24 hours or 1440 minutes,
if not then those values are invalid */
select * from
(
select VeryActiveMinutes, FairlyActiveMinutes, LightlyActiveMinutes, SedentaryMinutes,
(VeryActiveMinutes + FairlyActiveMinutes + LightlyActiveMinutes + SedentaryMinutes)
as total_minutes from dailyactivity_merged
  )
    dailyactivity_merged where total_minutes > 1440;
    -- No invalid values as the output was empty
```

4b: sleepday_merged table: Similar to step 4a, I tried to find the duplicate and null values in the sleepday_merged table and deleted those records if any. I found 3 duplicates. Since ids were also duplicated, I added a new column called 'row_num' to give each row a unique index, which made it easy to filter out and remove duplicates. I then crossed checked the accuracy of the data within this table in terms of whether all the activity minutes were adding up to 1440 minutes or not. No invalid values were found. As the date information also had time, I added a new column ‘date’ with just the date information.

```sql
/* finding duplicates in 'sleepday_merged' table */
select *, count(*) as duplicates from sleepday_merged
group by id, date_time, TotalSleepRecords, TotalMinutesAsleep, TotalTimeInBed
having count(*) > 1;            -- 3 Duplicates
```

```sql
/* Removing duplicates */
/* since ids are also duplicated I added a new column called 'row_num'
to give each row a unique identifier, which made it easy to filter out
and remove duplicates */
alter table sleepday_merged
add column row_num int auto_increment, add primary key (row_num);
```

```sql
/* deleting duplicates using 'row_num' column */
delete from sleepday_merged
where row_num in (select * from
        (select max(row_num) as rn
            from sleepday_merged
group by id, date_time, TotalSleepRecords, TotalMinutesAsleep, TotalTimeInBed
having count(*) > 1
                  ) sleepday_merged);
```

```sql
/* finding invalid values (values that are greater than 1440 mintues or 24 hours) */
select * from sleepday_merged
where TotalMinutesAsleep > 1440 or TotalTimeInBed > 1440;   -- NO invalid values
```

```sql
/* Adding a new column 'date' from 'date_time' */
alter table sleepday_merged;
-- add column date date after date_time;

update sleepday_merged
set date = date(date_time);
```

4c. hourlycalories_merged: Similar to step 4a and 4b, I tried to find the duplicate and null values in the hourlycalories_merged table and deleted those records if any. I did not find any duplicates or null values.

```sql
/* finding duplicates in 'hourlycalories_merged' table*/
select *, count(*) as duplicates from hourlycalories_merged
group by id, date_time, Calories having count(*) > 1;        -- No duplicates were found
```

```sql
/* identifying Missing/Null values */
select sum(case when id is null or id = 0 then 1 else 0 end) as missing_ids,
       sum(case when date_time is null or date_time = 0 then 1 else 0 end) as missing_dates,
       sum(case when Calories is null or Calories = 0 then 1 else 0 end) as missing_calories
from hourlycalories_merged;         -- No missing or Null values were found
```

4d. hourlyintensities_merged table: Similar to the above steps, I tried to find the duplicate and null values in the hourlyintensities_merged table and deleted those records if any. I did not find any duplicates, however, in total, '9097' missing values were found in both 'Totalintensity and AverageIntensity' columns.

```sql
/* finding duplicates in 'hourlyintensities_merged' table*/
select *, count(*) as duplicates from hourlyintensities_merged
group by id, date_time, TotalIntensity, AverageIntensity
having count(*) > 1;            -- No duplicates were found
```

```sql
/* identifying Missing/Null values */
/*select * from hourlyintensities_merged
where id is null  or id = 0
   or date_time is null or date_time = 0 or
      totalintensity is null or totalintensity = 0 or
      averageintensity is null or averageintensity = 0; */
select sum(case when id is null or id = 0 then 1 else 0 end)
as missing_ids,
       sum(case when date_time is null or date_time = 0 then 1 else 0 end)
       as missing_dates,
       sum(case when TotalIntensity is null or TotalIntensity = 0 then 1 else 0 end)
       as missing_intensities,
       sum(case when averageintensity is null or averageintensity = 0 then 1 else 0 end)
       as missing_avg_inensities
from hourlyintensities_merged;
-- '9097' missing values in both 'Totalintensity and AverageIntensity' columns
```

4e. hourlysteps_merged table: Lastly, I tried to find the duplicate and null values in the hourlysteps_merged table and deleted those records if any. I did not find any duplicates, however, in total, '9297' missing values were found in 'Steptotal' column.

```sql
/* finding duplicates in 'hourlysteps_merged' table */
select *, count(*) as duplicates
from hourlysteps_merged
group by id, datetime, StepTotal
having count(*) > 1;           -- No duplicates
```

```sql
/* identifying Missing/Null values */
select sum(case when id is null or id = 0 then 1 else 0 end) as missing_ids,
       sum(case when datetime is null or datetime = 0 then 1 else 0 end) as missing_dates,
       sum(case when steptotal is null or steptotal = 0 then 1 else 0 end) as missing_steptotal
from hourlysteps_merged;        -- '9297' missing values in 'Steptotal' column
```

4. Transforming the data and peforming further data analysis: In this step I combined the tables containing data related to daily activity (dailyactivity_merged, dailysleep_merged) into a new table called 'daily_activity_sleep' and tables containing hourly data (hourlycalories_merged, hourlyintensities_merged, hourlysteps_merged) into a new table called 'hourly_activity'. I also added a few columns in each table which would be useful to carry out analysis. For more details, refer to the SQL queries in the file here.

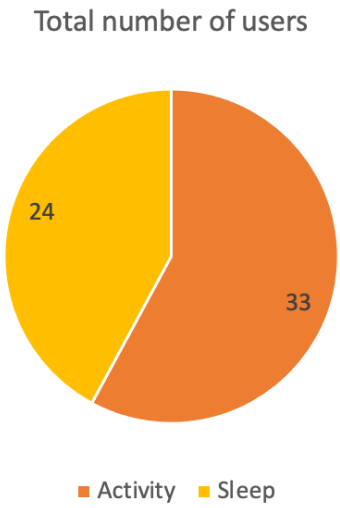
```sql
-- combining 'dailyactivity_merged' and 'dailysleep_merged'
create table daily_activity_sleep
select tbl1.*, tbl2.TotalSleepRecords, TotalMinutesAsleep, TotalTimeInBed
from dailyactivity_tbl  tbl1
join dailysleep_tbl     tbl2  using(id, date);
```

```sql
-- combining 'hourlycalories_merged' , 'hourlyintensities_merged' and 'hourlysteps_merged'
create table hourly_activity
select tbl1.*, tbl2.TotalIntensity, tbl2.AverageIntensity, tbl3.StepTotal
from hourlycalories_merged      tbl1
join hourlyintensities_merged   tbl2    using (id, date_time)
join hourlysteps_merged         tbl3    using (id, date_time);
```

5. Data analysis:

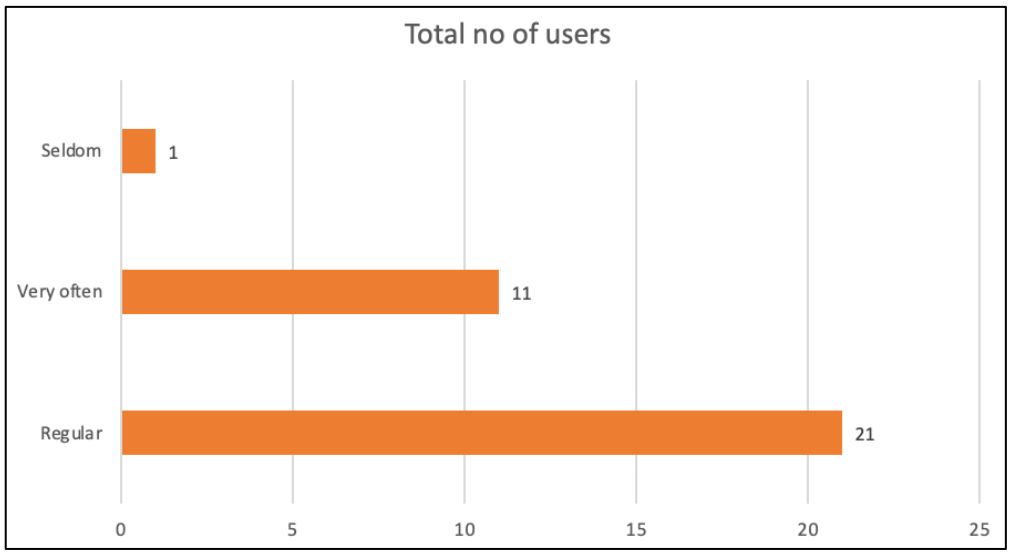
#1 Total users of each feature

For analysis, the pie chart below shows that in total 33 users used the activity tracker and only 24 users used the sleep tracker. It might have happened that the users were more comfortable in using the activity tracker as they were supposed to use it during the day time. However, the users were not comfortable to use the sleep tracker as it was supposed to be used at night. I contend that some individuals inherently don't prefer to use technology during the night time and stay away from devices. Hence the total number of users using sleep tracker were less compared to the activity tracker.



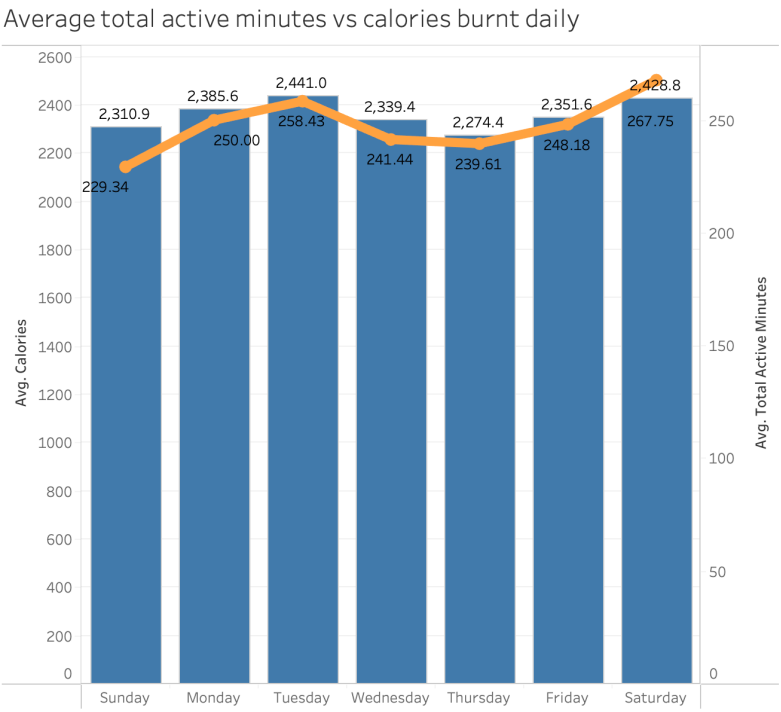
#2 Device usage level

The next thing I created a bar chart categorizing the users into different categories (seldom, very often and regular) based on their device usage level. I found that majority of the users came under the regular category (21-31 days each month, followed by very often (11-20 days) and then seldom (0-10 days). This shows that majority of the users might be using the tracker to achieve their fitness goals. It also highlights that maybe the 'very often' users need slightly more encouragement to become regular users of the tracker.



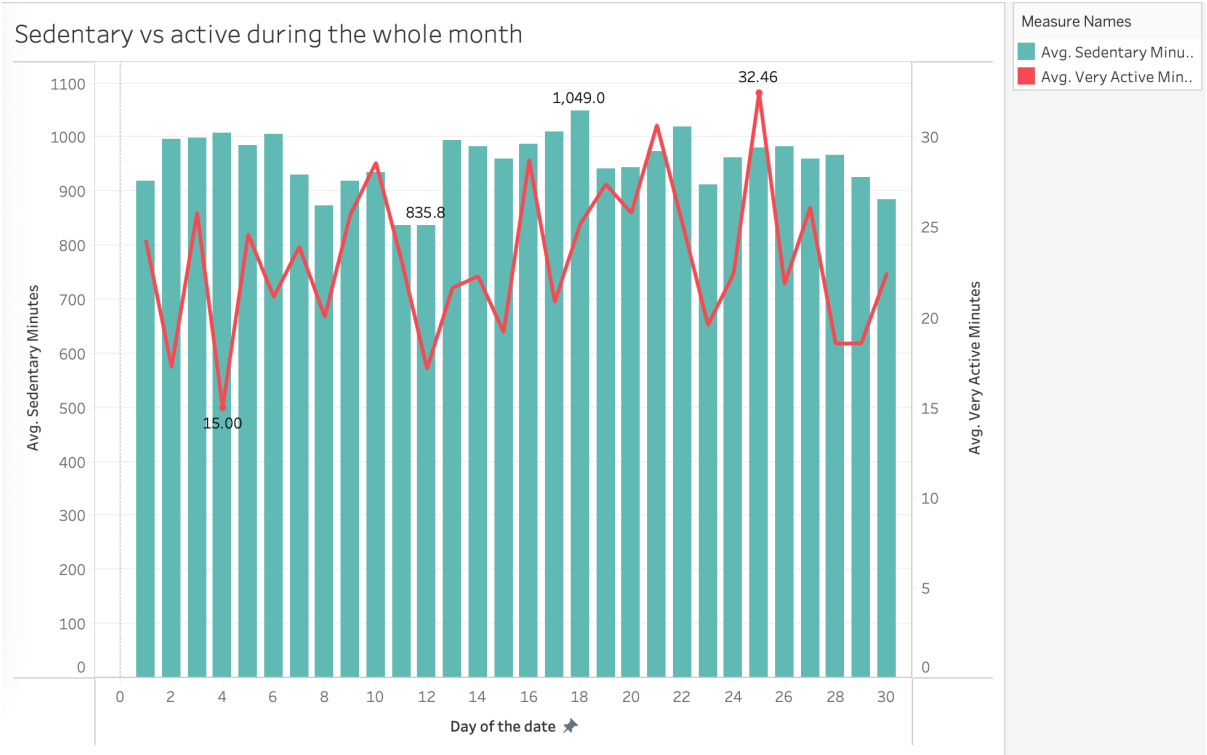
#3 User average active minutes during the week

The bar chart below shows that the users were the most active on Saturday and the least active on Sunday. Going further, I found that it matches with calories burnt on average during a particular day. This highlights that the tracker should encourage users by sending gentle reminders to be more active on their less active days, especially over the weekends.



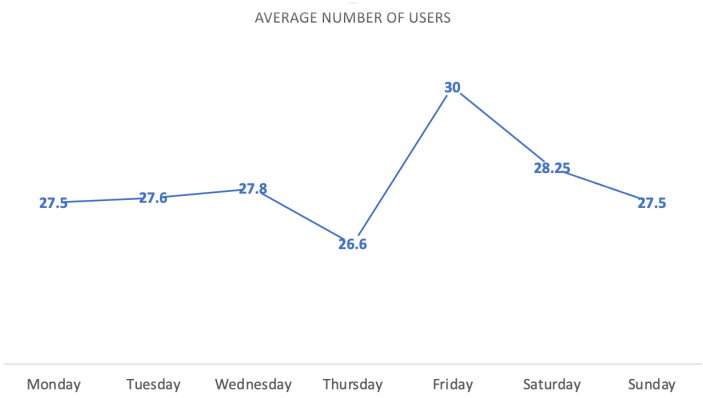
#4 sedentary vs active during the month

The bar chart below shows on average the days on which the user experienced being more active or more sedentary. In other words, this chart highlights the trend in user activity throughout the month. For instance, the users were seen to be the least active on day 4 of the month and and the most active on day 25 of the month.



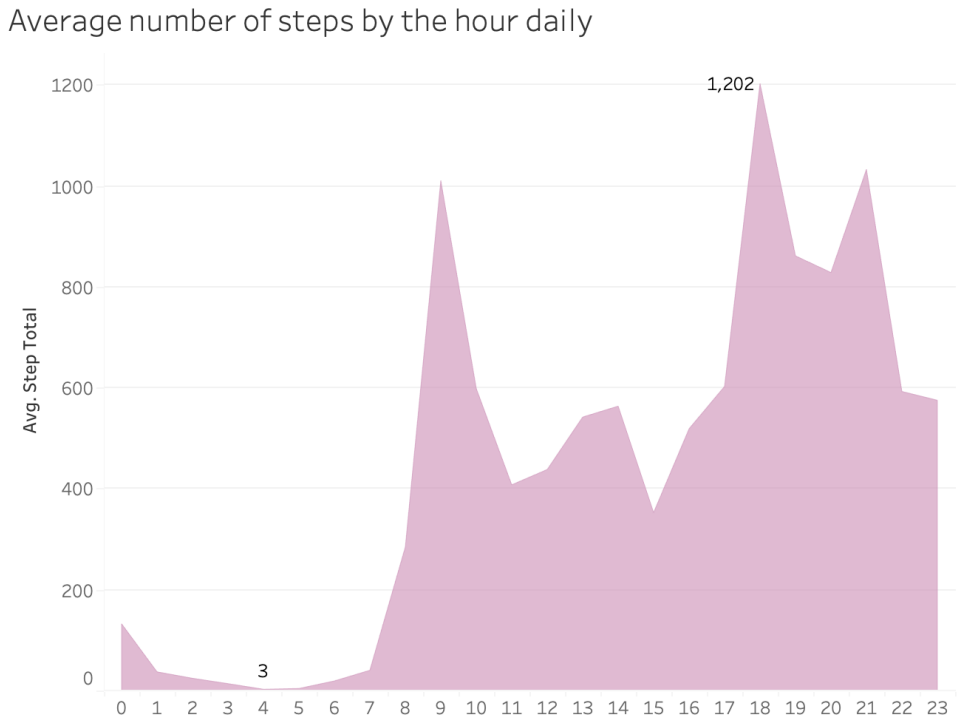
#5 usage rate by day of the week

Next, I examined the usage of the tracker over a week. I found that the device usage was the lowest on Thursdays and highest on Fridays. People might feel less motivated to use the device on Thursday because they've already been through most of the week and they might be exhausted. Fridays might see more usage because in the mind, users already foresee the weekend break and might want to hit their fitness goals before the week ends. Also, usually as the weekend is around, people might be wrapping up their work sooner on Fridays making them have lesser work compared to Thursdays.



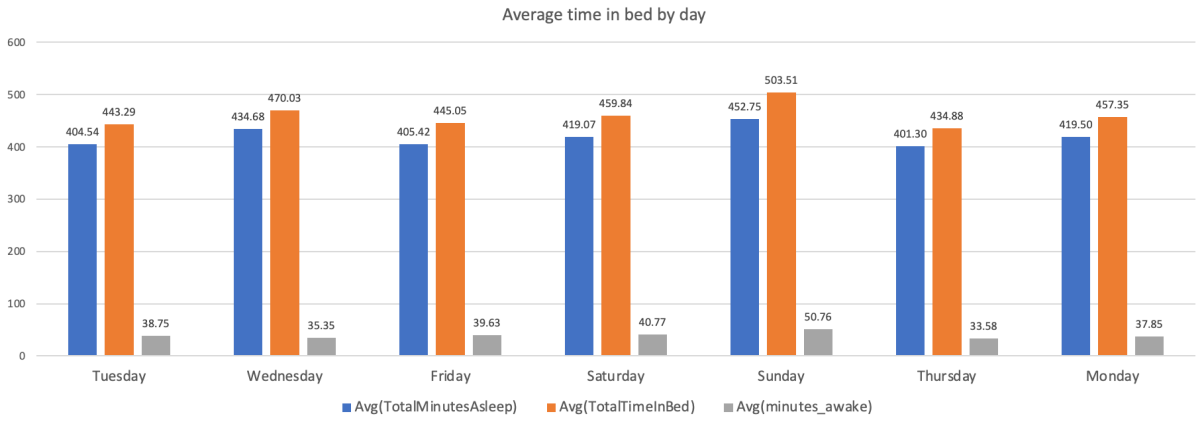
#6 Average steps taken by hour during the day

Looking at this graph, I found that on an average the users walked the most from 8 AM to 7 PM and the least from 12 AM to 4 AM. This shows they're more active during the day and less active at night. This is usual for people who have regular work hours.



#7 Average time in bed by day

The below graph shows that users have a stable sleep pattern throughout the week. However, they sleep more on Wednesdays and Sundays, possibly because Wednesday is the middle of the week and they have reached a certain level of exhaustion. And the users might be sleeping more on Sundays so that they are well rested before the week starts. they have less to do. I also found that they sleep less on Tuesdays, Thursdays, and Fridays, possibly because they might be juggling with loads of things during those days.



Phase 6: Act

After analyzing the FitBit Fitness Tracker data, I came up with some recommendations for Bellabeat marketing strategy based on trends in smart device usage.

1. Majority of the participants were lightly active. Bellabeat should offer a pop-up notification feature to encourage user to level up in small steps (let's say by 2000 steps each time to reach a goal of 10,000 steps) and get extra rewards/ points if they did so. This will help the participants to become at least fairly active with the aim of reaching very active level eventually.
2. Bellabeat could also consider providing some activity suggestions as per the initially selected user preference that will help the user to reduce their sedentary time.
3. Bellabeat could consider expanding the scope of their features and also provide users some ideas for low/ high calorie breakfast, lunch, and dinner foods to help users that wish to manage their body weight. This can help different kinds of users with different weight goals such as losing or gaining weight.
4. It was found that there were lesser number of users for sleep tracker as compared to the other feature. So, Bellabeat could consider using daily pop-up notifications reminding users to get enough rest. Even though many users prefer the Activity Tracker, paying attention to sleep patterns is crucial for overall health management. To encourage more Leaf users, Bellabeat could consider adding useful features such as sleep tips, quality analysis, or smart alarms to the Sleep Tracker.
5. Users were found to be the most active on Saturdays. Bellabeat could use this insight to remind users or suggest users to go for a walk, a jog or do some sort to activity on that day. Participants seem to be the least active on Sundays. Bellabeat can use this to

appreciate users by sending them motivational pop ups and also encourage them to keep being involved in some sort of activity and continue exercising on Sundays.

Note: That completes my Bellabeat case study. Moreover, this data has some limitations. With whatever information was given in the dataset, I have come up with some recommendations