# INTRODUCTION

This case study is the Capstone Project of **Google Data Analytics Professional Certificate**. The **6 steps of Data Analysis** is used to present this analysis.

Title: **Bellabeat Case Study**

Author: Katie Huang

Date: 2 June 2021

Logo

Description automatically generated with medium confidence

## Bellabeat: How Can A Wellness Technology Company Play It Smart?

# STEP 1: ASK

#### 1.0 Background

Bellabeat is a high-tech manufacturer of beautifully-designed health-focused smart products for women since 2013. Inspiring and empowering women with knowledge about their own health and habits, Bellabeat has grown rapidly and quickly positioned itself as a tech-driven wellness company for females.

The co-founder and Chief Creative Officer, Urška Sršen is confident that an analysis of non-Bellebeat consumer data (ie. FitBit fitness tracker usage data) would reveal more opportunities for growth.

#### 1.2 Business Task:

Analyze FitBit Fitness Tracker Data to gain insights into how consumers are using the FitBit app and discover trends and insights for Bellabeat marketing strategy.

#### 1.3 Business Objectives:

1. What are the trends identified?
2. How could these trends apply to Bellabeat customers?
3. How could these trends help influence Bellabeat marketing strategy?

#### 1.4 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 analysis
5. Supporting visualizations and key findings
6. High-level content recommendations based on the analysis

#### 1.5 Key Stakeholders:

1. Urška Sršen: Bellabeat’s cofounder and Chief Creative Officer
2. Sando Mur: Mathematician, Bellabeat’s cofounder and key member of the Bellabeat executive team
3. Bellabeat marketing analytics team: A team of data analysts guiding Bellabeat's marketing strategy.

# STEP 2: PREPARE

#### 2.1 Information on Data Source:

1. The data is publicly available on [Kaggle: FitBit Fitness Tracker Data](https://www.kaggle.com/arashnic/fitbit) and stored in 18 csv files.
2. Generated by respondents from a distributed survey via Amazon Mechanical Turk between 12 March 2016 to 12 May 2016.
3. 30 FitBit users who consented to the submission of personal tracker data.
4. Data collected includes (1) physical activity recorded in minutes, (2) heart rate, (3) sleep monitoring, (4) daily activity and (5) steps.

#### 2.2 Limitations of Data Set:

1. Data collected from year 2016. Users' daily activity, fitness and sleeping habits, diet and food consumption may have changed since then, hence data may not be timely or relevant.
2. Sample size of 30 female FitBit users is not representative of the entire female population.
3. As data is collected in a survey, hence unable to ascertain the integrity or accuracy of data.

#### 2.3 Is Data ROCCC?

A good data source is ROCCC which stands for **R**eliable, **O**riginal, **C**omprehensive, **C**urrent, and **C**ited.

1. Reliable - LOW - Not reliable as it only has 30 respondents
2. Original - LOW - Third party provider (Amazon Mechanical Turk)
3. Comprehensive - MED - Parameters match most of Bellabeat's products' parameters
4. Current - LOW - Data is 5 years old and is not relevant
5. Cited - LOW - Data collected from third party, hence unknown

Overall, the dataset is considered bad quality data and it is not recommended to produce business recommendations based on this data.

#### 2.4 Data Selection:

The following file is selected and copied for analysis.

• dailyActivity\_merged.csv

# STEP 3: PROCESS

We are using Python to prepare and process the data.

3.1 Preparing the Environment

The numPy, pandas, matplotlib, datetime packages are installed and aliased for easy reading.

*# import packages and alias*

import numpy as np *# data arrays*

import pandas as pd *# data structure and data analysis*

import matplotlib as plt *# data visualization*

import datetime as dt *# date time*

#### 3.2 Importing data set

Reading in the selected file.

*# read\_csv function to read the required CSV file*

daily\_activity = pd.read\_csv("../input/fitbit/Fitabase Data 4.12.16-5.12.16/dailyActivity\_merged.csv")

linkcode

#### 3.3 Data cleaning and manipulation

##### ***Steps***

1. Observe and familiarize with data
2. Check for null or missing values
3. Perform sanity check of data

Previewing using head function to show the first 10 rows of daily\_activity to familiarise with the data.

daily\_activity.head(10)

Then, finding out whether there is any null or missing values in daily\_activity.

*# obtain the # of missing data points per column*

missing\_values\_count = daily\_activity.isnull().sum()

*# look at the # of missing points in all columns*

missing\_values\_count[:]

Finding out the basic information of daily\_activity:

* no. of rows and columns
* name of columns
* type of value

*# show basic information of data*

daily\_activity.info()

Counting the unique ID and to confirm whether data set has 30 IDs.

*# count distinct value of "Id"*

unique\_id = len(pd.unique(daily\_activity["Id"]))

print("# of unique Id: " + str(unique\_id))

From the above observation, noted that

1. There is no typo, Null or missing values.
2. Data frame has 940 rows and 15 columns.
3. *ActivityDate* is wrongly classified as object dtype and has to be converted to datetime64 dtype.
4. There are 33 unique IDs, instead of 30 unique IDs as expected from 30 fitness tracker users.

The following data manipulation is performed:

1. Convert *ActivityDate* to datatime64 dtype.
2. Convert format of ActivityDate to yyyy-mm-dd.
3. Create new column DayOfTheWeek by separating the date into day of the week for further analysis.
4. Create new column TotalMins being the sum of VeryActiveMinutes, FairlyActiveMinutes, LightlyActiveMinutes and SedentaryMinutes.
5. Create new column TotalHours by converting new column in #4 to number of hours.
6. Rearrange and rename columns.

Converting *ActivityDate* from object to datatime64 dtype and converting format of *ActivityDate* to yyyy-mm-dd. Then, printing head to confirm whether it has been updated to datatime64 dtype and dates to yyyy-mm-dd.

*# convert "ActivityDate" to datatime64 dtype and format to yyyy-mm-dd*

daily\_activity["ActivityDate"] = pd.to\_datetime(daily\_activity["ActivityDate"], format="%m/**%d**/%Y")

*# re-print information to confirm*

daily\_activity.info()

*# print the first 5 rows of "ActivityDate" to confirm*

daily\_activity["ActivityDate"].head(5)

*#r create new list of rearranged columns*

new\_cols = ['Id', 'ActivityDate', 'DayOfTheWeek', 'TotalSteps', 'TotalDistance', 'TrackerDistance', 'LoggedActivitiesDistance', 'VeryActiveDistance', 'ModeratelyActiveDistance', 'LightActiveDistance', 'SedentaryActiveDistance', 'VeryActiveMinutes', 'FairlyActiveMinutes', 'LightlyActiveMinutes', 'SedentaryMinutes', 'TotalExerciseMinutes', 'TotalExerciseHours', 'Calories']

*# reindex function to rearrange columns based on "new\_cols"*

df\_activity = daily\_activity.reindex(columns=new\_cols)

*# print 1st 5 rows to confirm*

df\_activity.head(5)

Creating new column by separating the date into day of the week for further analysis.

*# create new column "day\_of\_the\_week" to represent day of the week*

df\_activity["DayOfTheWeek"] = df\_activity["ActivityDate"].dt.day\_name()

*# print 1st 5 rows to confirm*

df\_activity["DayOfTheWeek"].head(5)

Rearranging and renaming columns from XxxYyy to xxx\_yyy.

*# rename columns*

df\_activity.rename(columns = {"Id":"id", "ActivityDate":"date", "DayOfTheWeek":"day\_of\_the\_week", "TotalSteps":"total\_steps", "TotalDistance":"total\_dist", "TrackerDistance":"track\_dist", "LoggedActivitiesDistance":"logged\_dist", "VeryActiveDistance":"very\_active\_dist", "ModeratelyActiveDistance":"moderate\_active\_dist", "LightActiveDistance":"light\_active\_dist", "SedentaryActiveDistance":"sedentary\_active\_dist", "VeryActiveMinutes":"very\_active\_mins", "FairlyActiveMinutes":"fairly\_active\_mins", "LightlyActiveMinutes":"lightly\_active\_mins", "SedentaryMinutes":"sedentary\_mins", "TotalExerciseMinutes":"total\_mins","TotalExerciseHours":"total\_hours","Calories":"calories"}, inplace = True)

*# print column names to confirm*

print(df\_activity.columns.values)

df\_activity.head(5)

Creating new column *total\_mins* being the sum of total time logged.

*# create new column "total\_mins" containing sum of total minutes.*

df\_activity["total\_mins"] = df\_activity["very\_active\_mins"] + df\_activity["fairly\_active\_mins"] + df\_activity["lightly\_active\_mins"] + df\_activity["sedentary\_mins"]

df\_activity["total\_mins"].head(5)

Creating new column by converting *total\_mins* to number of hours.

*# create new column \*total\_hours\* by converting to hour and round float to two decimal places*

df\_activity["total\_hours"] = round(df\_activity["total\_mins"] / 60)

*# print 1st 5 rows to confirm*

df\_activity["total\_hours"].head(5)

Data cleaning and manipulation is completed. Hence, data is now ready to be analysed.

# **STEP 4: ANALYZE**

#### 4.1 Perform calculations

Pulling the statistics of df\_activity for analysis:

* count - no. of rows
* mean (average)
* std (standard deviation)
* min and max
* percentiles 25%, 50%, 75%

*# pull general statistics*

df\_activity.describe()

Interpreting statistical findings:

1. On average, users logged 7,637 steps or 5.4km which is not adequate. As recommended by CDC, an adult female has to aim at least 10,000 steps or 8km per day to benefit from general health, weight loss and fitness improvement.
2. Sedentary users are the majority logging on average 991 minutes or 20 hours making up 81% of total average minutes.
3. Noting that average calories burned is 2,303 calories equivalent to 0.6 pound. Could not interpret into detail as calories burned depend on several factors such as the age, weight, daily tasks, exercise, hormones and daily calorie intake.

# **STEP 5: SHARE**

In this step, we are creating visualizations and communicating our findings based on our analysis.

#### 5.1 Data Visualisation and Findings

*# import matplotlib package*

import matplotlib.pyplot as plt

*# plotting histogram*

plt.style.use("default")

plt.figure(figsize=(6,4)) *# specify size of the chart*

plt.hist(df\_activity.day\_of\_the\_week, bins = 7,

width = 0.6, color = "lightskyblue", edgecolor = "black")

*# adding annotations and visuals*

plt.xlabel("Day of the week")

plt.ylabel("Frequency")

plt.title("No. of times users logged in app across the week")

plt.grid(True)

plt.show()

Chart, bar chart

Description automatically generated

##### ***Frequency of usage across the week***

In this histogram, we are looking at the frequency of FitBit app usage in terms of days of the week.

1. We discovered that users prefer or remember (giving them the doubt of benefit that they forgotten) to track their activity on the app during midweek from Tuesday to Friday.
2. Noting that the frequency dropped on Friday and continue weekends and Monday.

*# import matplotlib package*

import matplotlib.pyplot as plt

*# plotting scatter plot*

plt.style.use("default")

plt.figure(figsize=(8,6)) *# specify size of the chart*

plt.scatter(df\_activity.total\_steps, df\_activity.calories,

alpha = 0.8, c = df\_activity.calories,

cmap = "Spectral")

*# add annotations and visuals*

median\_calories = 2303

median\_steps = 7637

plt.colorbar(orientation = "vertical")

plt.axvline(median\_steps, color = "Blue", label = "Median steps")

plt.axhline(median\_calories, color = "Red", label = "Median calories burned")

plt.xlabel("Steps taken")

plt.ylabel("Calories burned")

plt.title("Calories burned for every step taken")

plt.grid(True)

plt.legend()

plt.show()

Chart, scatter chart

Description automatically generated

##### ***Calories burned for every step taken***

From the scatter plot, we discovered that:

1. It is a positive correlation.
2. We observed that intensity of calories burned increase when users are at the range of > 0 to 15,000 steps with calories burn rate cooling down from 15,000 steps onwards.
3. Noted a few outliers:
   * Zero steps with zero to minimal calories burned.
   * 1 observation of > 35,000 steps with < 3,000 calories burned.
   * Deduced that outliers could be due to natural variation of data, change in user's usage or errors in data collection (ie. miscalculations, data contamination or human error).

*# import matplotlib package*

import matplotlib.pyplot as plt

*# plotting scatter plot*

plt.style.use("default")

plt.figure(figsize=(8,6)) *# Specify size of the chart*

plt.scatter(df\_activity.total\_hours, df\_activity.calories,

alpha = 0.8, c = df\_activity.calories,

cmap = "Spectral")

*# adding annotations and visuals*

median\_calories = 2303

median\_hours = 20

median\_sedentary = 991 / 60

plt.colorbar(orientation = "vertical")

plt.axvline(median\_hours, color = "Blue", label = "Median steps")

plt.axvline(median\_sedentary, color = "Purple", label = "Median sedentary")

plt.axhline(median\_calories, color = "Red", label = "Median hours")

plt.xlabel("Hours logged")

plt.ylabel("Calories burned")

plt.title("Calories burned for every hour logged")

plt.legend()

plt.grid(True)

plt.show()

Chart, scatter chart

Description automatically generated

##### ***Calories burned for every hour logged***

The scatter plot is showing:

1. A weak positive correlation whereby the increase of hours logged does not translate to more calories being burned. That is largely due to the average sedentary hours (purple line) plotted at the 16 to 17 hours range.
2. Again, we can see a few outliers:
   * The same zero value outliers
   * An unusual red dot at the 24 hours with zero calorie burned which may be due to the same reasons as above.

*# import packages*

import matplotlib.pyplot as plt

import numpy as np

*# calculating total of individual minutes column*

very\_active\_mins = df\_activity["very\_active\_mins"].sum()

fairly\_active\_mins = df\_activity["fairly\_active\_mins"].sum()

lightly\_active\_mins = df\_activity["lightly\_active\_mins"].sum()

sedentary\_mins = df\_activity["sedentary\_mins"].sum()

*# plotting pie chart*

slices = [very\_active\_mins, fairly\_active\_mins, lightly\_active\_mins, sedentary\_mins]

labels = ["Very active minutes", "Fairly active minutes", "Lightly active minutes", "Sedentary minutes"]

colours = ["lightcoral", "yellowgreen", "lightskyblue", "darkorange"]

explode = [0, 0, 0, 0.1]

plt.style.use("default")

plt.pie(slices, labels = labels,

colors = colours, wedgeprops = {"edgecolor": "black"},

explode = explode, autopct = "**%1.1f%%**")

plt.title("Percentage of Activity in Minutes")

plt.tight\_layout()

plt.show()

Chart, pie chart

Description automatically generated

**Percentage of Activity in Minutes**

As seen from the pie chart,

1. Sedentary minutes takes the biggest slice at 81.3%.
2. This indicates that users are using the FitBit app to log daily activities such as daily commute, inactive movements (moving from one spot to another) or running errands.
3. App is rarely being used to track fitness (ie. running) as per the minor percentage of fairly active activity (1.1%) and very active activity (1.7%). This is highly discouraging as FitBit app was developed to encourage fitness.

# **STEP 6: ACT**

In the final step, we will be delivering our insights and providing recommendations based on our analysis.

Here, we revisit our business questions and share with you our high-level business recommendations.

**1. What are the trends identified?**

* Majority of users (81.3%) are using the FitBit app to track sedentary activities and not using it for tracking their health habits.
* Users prefer to track their activities during weekdays as compared to weekends - perhaps because they spend more time outside on weekdays and stay in on weekends.

**2. How could these trends apply to Bellabeat customers?**

* Both companies develop products focused on providing women with their health, habit and fitness data and encouraging them to understand their current habits and make healthy decisions. These common trends surrounding health and fitness can very well applied to Bellabeat customers.

**3. How could these trends help influence Bellabeat marketing strategy?**

* Bellabeat marketing team can encourage users by educating and equipping them with knowledge about fitness benefits, suggest different types of exercise (ie. simple 10 minutes exercise on weekday and a more intense exercise on weekends) and calories intake and burnt rate information on the Bellabeat app.
* On weekends, Bellabeat app can also prompt notification to encourage users to exercise.