Google Data Analytics Capstone Project

Case Study 2:How can a wellness technology company play it smart?

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. 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.

Key Stakeholders

- Urška Sršen: Cofounder and Chief Creative Officer
- Sando Mur: Cofounder and Mathematician

Business Task

Analyze smart device usage data in order to gain insight into how consumers use non-Bellabeat smart devices, with focus on a Bellabeat product and guiding my analysis with these questions:

- 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?

The dataset used for this analysis is the FitBit Fitness Tracker Data public data, made available through Möbius.

loading the data

we would go ahead and set up our working directory as well as import the csv files to be used for our analysis.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.3     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
```

```
## Warning: package 'readr' was built under R version 4.1.1
```

```
## -- Conflicts --------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(knitr)
library(rmarkdown)
setwd("/Users/nancy/downloads/names")
daily_activity <- read_csv("Fitabase Data 4.12.16-5.12.16/dailyActivity_merged.csv")
```

```
## Rows: 940 Columns: 15
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): ActivityDate
## dbl (14): Id, TotalSteps, TotalDistance, TrackerDistance, LoggedActivitiesDi...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

next, we explore our dataset and make observations on the various rows and columns.

```
head(daily_activity)
```

```
## # A tibble: 6 x 15
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance LoggedActivitie~
##        <dbl> <chr>             <dbl>         <dbl>           <dbl>             <dbl>
## 1 1503960366 4/12/2016         13162          8.5             8.5                 0
## 2 1503960366 4/13/2016         10735          6.97            6.97                0
## 3 1503960366 4/14/2016         10460          6.74            6.74                0
## 4 1503960366 4/15/2016          9762          6.28            6.28                0
## 5 1503960366 4/16/2016         12669          8.16            8.16                0
## 6 1503960366 4/17/2016          9705          6.48            6.48                0
## # ... with 9 more variables: VeryActiveDistance <dbl>,
## #   ModeratelyActiveDistance <dbl>, LightActiveDistance <dbl>,
## #   SedentaryActiveDistance <dbl>, VeryActiveMinutes <dbl>,
## #   FairlyActiveMinutes <dbl>, LightlyActiveMinutes <dbl>,
## #   SedentaryMinutes <dbl>, Calories <dbl>
```

we can see that some of our columns consist of only one value (0), this is irrelevant to our analysis, we can select the relevant columns to a new dataframe for further cleaning and drop the columns we do not need using the select() and subset() function.

```
clean_daily_activity <- subset(daily_activity, select = -c(SedentaryActiveDistance, LoggedActivitiesDis

#confirming our dropped columns
head(clean_daily_activity)
```

```
## # A tibble: 6 x 13
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance VeryActiveDista~
##        <dbl> <chr>             <dbl>         <dbl>           <dbl>            <dbl>
## 1 1503960366 4/12/2016         13162          8.5             8.5              1.88
## 2 1503960366 4/13/2016         10735          6.97            6.97             1.57
## 3 1503960366 4/14/2016         10460          6.74            6.74             2.44
## 4 1503960366 4/15/2016          9762          6.28            6.28             2.14
## 5 1503960366 4/16/2016         12669          8.16            8.16             2.71
## 6 1503960366 4/17/2016          9705          6.48            6.48             3.19
## # ... with 7 more variables: ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, VeryActiveMinutes <dbl>,
## #   FairlyActiveMinutes <dbl>, LightlyActiveMinutes <dbl>,
## #   SedentaryMinutes <dbl>, Calories <dbl>
```

cleaning the dataset

we will explore this dataset further to find irregularities, clean the columns or create new columns. First, we convert our ActivityDate column from the character to date class:

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
clean_daily_activity$ActivityDate <- as.Date(clean_daily_activity$ActivityDate,format = "%m/%d/%Y")
head(clean_daily_activity)
```

```
## # A tibble: 6 x 13
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance VeryActiveDista~
##        <dbl> <date>            <dbl>         <dbl>           <dbl>            <dbl>
## 1 1503960366 2016-04-12        13162          8.5             8.5              1.88
## 2 1503960366 2016-04-13        10735          6.97            6.97             1.57
## 3 1503960366 2016-04-14        10460          6.74            6.74             2.44
## 4 1503960366 2016-04-15         9762          6.28            6.28             2.14
## 5 1503960366 2016-04-16        12669          8.16            8.16             2.71
## 6 1503960366 2016-04-17         9705          6.48            6.48             3.19
## # ... with 7 more variables: ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, VeryActiveMinutes <dbl>,
## #   FairlyActiveMinutes <dbl>, LightlyActiveMinutes <dbl>,
## #   SedentaryMinutes <dbl>, Calories <dbl>
```

now, let's create a new column with the active days of the week, we are going to extract the weekday from the ActivityDate column using the weekdays() function:

```
# Adding a new column for active days of the week
clean_daily_activity$Activeday <- weekdays(clean_daily_activity$ActivityDate)
head(clean_daily_activity)
```

```
## # A tibble: 6 x 14
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance VeryActiveDista~
##        <dbl> <date>            <dbl>         <dbl>           <dbl>            <dbl>
## 1 1503960366 2016-04-12        13162          8.5             8.5              1.88
## 2 1503960366 2016-04-13        10735          6.97            6.97             1.57
## 3 1503960366 2016-04-14        10460          6.74            6.74             2.44
## 4 1503960366 2016-04-15         9762          6.28            6.28             2.14
## 5 1503960366 2016-04-16        12669          8.16            8.16             2.71
## 6 1503960366 2016-04-17         9705          6.48            6.48             3.19
## # ... with 8 more variables: ModeratelyActiveDistance <dbl>,
## #   LightActiveDistance <dbl>, VeryActiveMinutes <dbl>,
## #   FairlyActiveMinutes <dbl>, LightlyActiveMinutes <dbl>,
## #   SedentaryMinutes <dbl>, Calories <dbl>, Activeday <chr>
```

our active day column has been created. Next, we are going to create a column ranking the total steps daily, we would perform this ranking based on the activity levels outlined in this article by CQUniversity Australia.

```
clean_daily_activity <- clean_daily_activity %>%
  mutate(clean_daily_activity, step_rank = case_when(
  TotalSteps < 100 ~ "N/A",
  TotalSteps >= 100 & TotalSteps < 5000 ~ "sedentary",
  TotalSteps >= 5000 & TotalSteps < 7500 ~ "low active",
  TotalSteps >= 7500 & TotalSteps < 10000 ~ "somewhat active",
  TotalSteps >= 10000 & TotalSteps < 12500 ~ "active",
  TotalSteps >= 12500 ~ "highly active"))
```

let us go ahead and check the number of N/A values in the step_rank column. We will make use of the table() function to check the frequency of each rank.

```
table(clean_daily_activity$step_rank)
```

```
##
##          active   highly active      low active             N/A       sedentary
##             159             144             171              87             216
## somewhat active
##             163
```

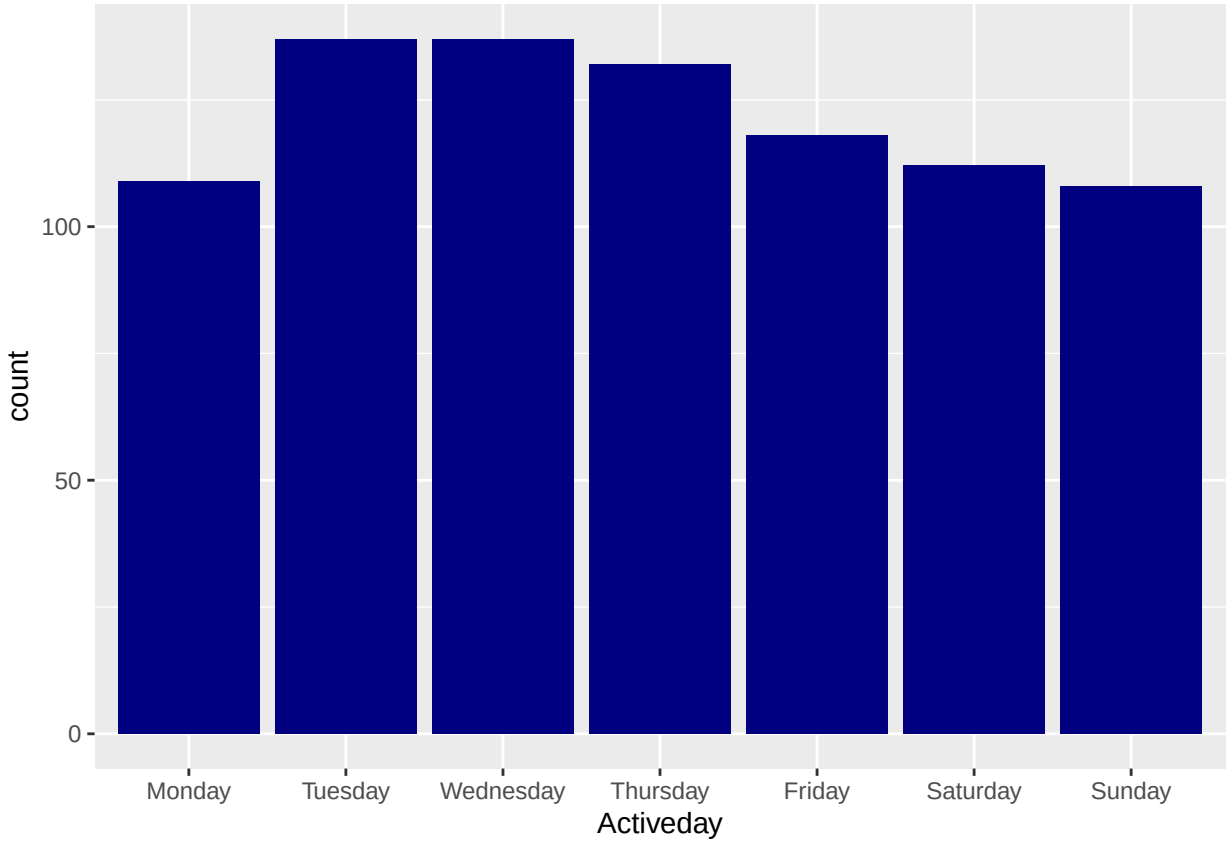
"N/A" has 87 values, we would go ahead and filter them out as these values would be outliers.

```
# filtering out the "N/A" values in the step_rank column
clean_daily_activity <- clean_daily_activity %>%
  filter(step_rank != "N/A")

clean_daily_activity$Activeday <- ordered(clean_daily_activity$Activeday, levels=c("Monday", "Tuesday",

clean_daily_activity$step_rank <- ordered(clean_daily_activity$step_rank, levels=c("sedentary", "low ac
```

Analyze and visualize our data

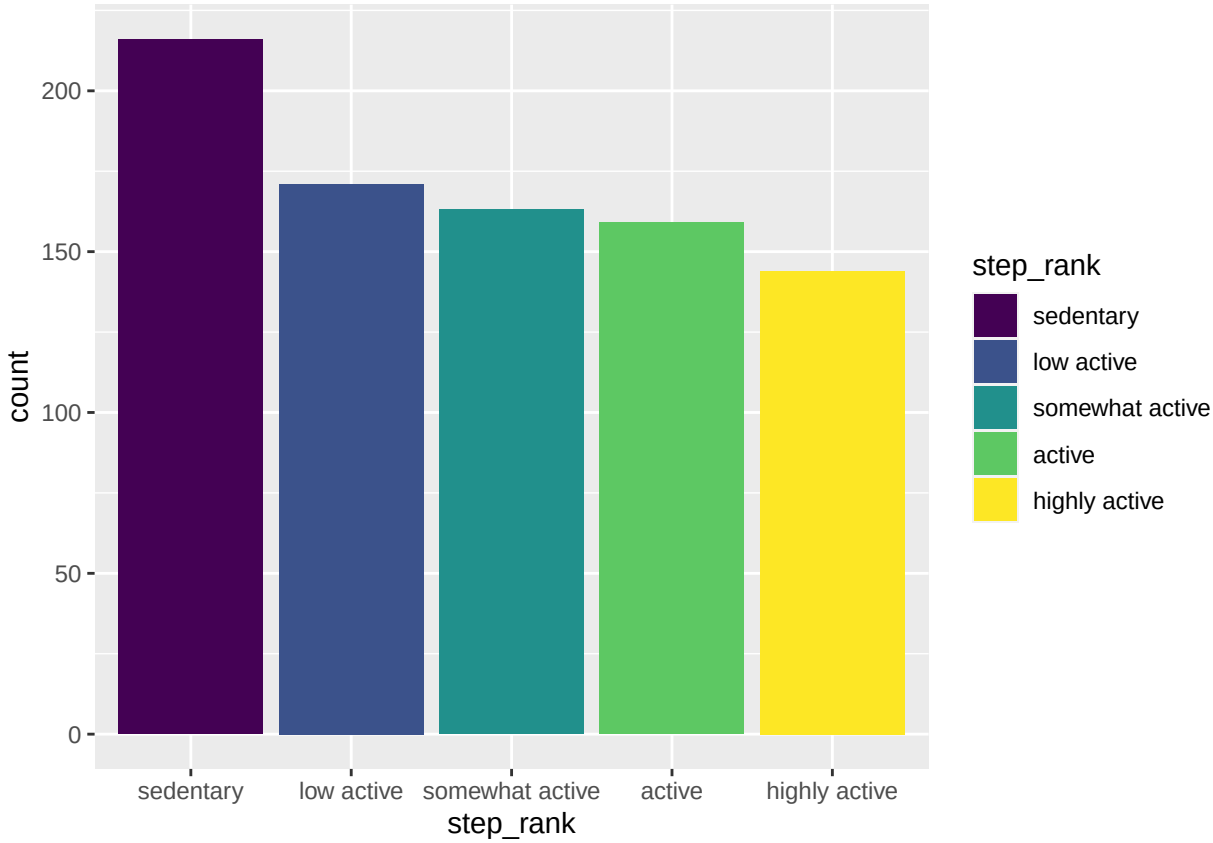
Our first visualization would be a count of the active days of the week to ascertain if there is a weekday/weekend pattern:

```
ggplot(data = clean_daily_activity) +
  geom_bar(mapping = aes(x=Activeday), fill="navyblue")
```



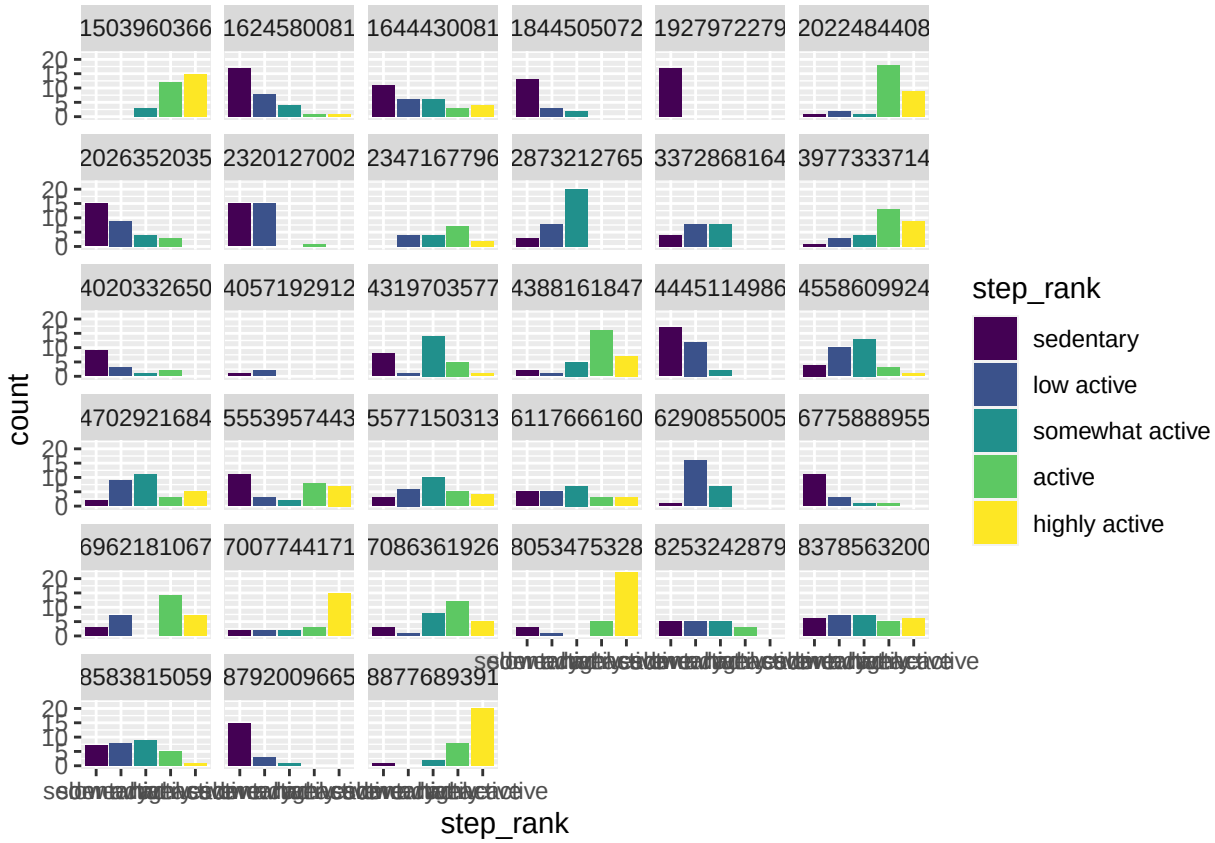
from the active days, we can see that people tend to be more active during the week, maybe employed people or students, our target demography therefore should be people between the ages 18-50, as most students and workers fall within this age range. next, let us make a bar chart of our step rank and see how active smart device users are:

```
ggplot(data = clean_daily_activity) +
  geom_bar(mapping = aes(x=step_rank, fill=step_rank))
```



from our chart, we can see that sedentary(a way of life characterized by much sitting and little exercise) is ranked the most,we are going to plot this again but this time, match it to each individual or unique ID to see if this step rank is the same for each person:

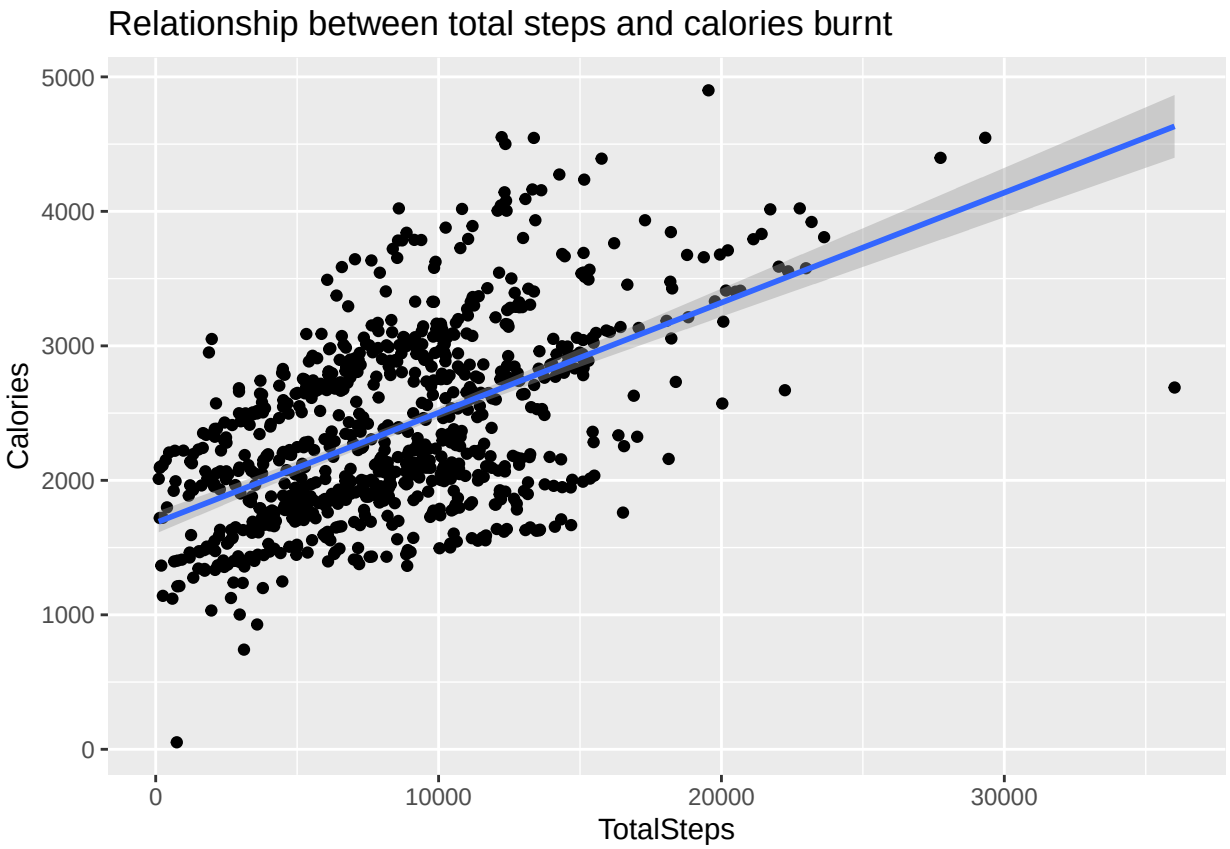
```
ggplot(data = clean_daily_activity) +
  geom_bar(mapping = aes(x=step_rank, fill=step_rank))+
  facet_wrap(~Id)
```



we can see that our individual charts are far from uniform and the sedentary ranking is high in some and very low in others, a good suggestion here would be to tailor our product,specifically the Bellabeat's Time, to fit individual needs and lifestyles, probably a timer or prompt for people working from home at different times during the day to take a brisk walks.

let us explore the total steps and calories columns to see if there is any correlation between steps taken in a day and calories burnt those days as well.

```
clean_daily_activity %>%
  ggplot(aes(x=TotalSteps,y=Calories)) + geom_point() + geom_smooth(method = "lm", formula=y~x) + labs(
```



From our scatterplot, we see that there is a positive correlation between total steps and calories, so the more active a product user is, the more calories they burn, this could be a good point for marketing the need for Bellabeat's Time.

Recommendations and Conclusion

- Bellabeat's target market should be the demography that is deemed the most active(18-50)
- product should be tailored to each users profile to find what works for the and how best to utilize the product.

In conclusion, Bellabeat has a large market of the active population trying to adopt healthier lifestyles, giving the user a personalized feel of the product should help acquire more users.