Bellabeat

Brunda Suresh

8/6/2021

After working on the cyclistic Case study using the roadmap provided by the course, I was intrested in conducting another case study . I went ahead and completed this case study !!

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, cofounder 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.

Sršen knows that an analysis of Bellabeat's available consumer data would reveal more opportunities for growth. She 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.

```
library(tidyverse)
```

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

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

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

```
library(lubridate)
```

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

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

```
library(ggplot2)
```

I have downloaded from kaggle, the bellabeat dataset consisting of customer tracker data between April 12th 2016 to May 12th 2016. I have used 3 of the csv files for my case study.

```
daily_activity<-read_csv("Fitabase_Data/dailyActivity_merged.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Id = col_double(),
##   ActivityDate = col_character(),
##   TotalSteps = col_double(),
##   TotalDistance = col_double(),
##   TrackerDistance = col_double(),
##   LoggedActivitiesDistance = col_double(),
##   VeryActiveDistance = col_double(),
##   ModeratelyActiveDistance = col_double(),
##   LightActiveDistance = col_double(),
##   SedentaryActiveDistance = col_double(),
##   VeryActiveMinutes = col_double(),
##   FairlyActiveMinutes = col_double(),
##   LightlyActiveMinutes = col_double(),
##   SedentaryMinutes = col_double(),
##   Calories = col_double()
## )
```

```
daily_steps<-read_csv("Fitabase_Data/dailySteps_merged.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Id = col_double(),
##   ActivityDay = col_character(),
##   StepTotal = col_double()
## )
```

```
heart_rate<-read_csv("Fitabase_Data/heartrate_seconds_merged.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Id = col_double(),
##   Time = col_character(),
##   Value = col_double()
## )
```

In the column specifications we see that the Date columns are in Character format and are to be changed. I continued by using the summary function to understand the the data at hand better.

```r
summary(daily_activity)
```

```
##        Id             ActivityDate        TotalSteps    TotalDistance
##  Min.   :1.504e+09   Length:940         Min.   :    0   Min.   : 0.000
##  1st Qu.:2.320e+09   Class :character   1st Qu.: 3790   1st Qu.: 2.620
##  Median :4.445e+09   Mode  :character   Median : 7406   Median : 5.245
##  Mean   :4.855e+09                      Mean   : 7638   Mean   : 5.490
##  3rd Qu.:6.962e+09                      3rd Qu.:10727   3rd Qu.: 7.713
##  Max.   :8.878e+09                      Max.   :36019   Max.   :28.030
##  TrackerDistance  LoggedActivitiesDistance VeryActiveDistance
##  Min.   : 0.000   Min.   :0.0000           Min.   : 0.000
##  1st Qu.: 2.620   1st Qu.:0.0000           1st Qu.: 0.000
##  Median : 5.245   Median :0.0000           Median : 0.210
##  Mean   : 5.475   Mean   :0.1082           Mean   : 1.503
##  3rd Qu.: 7.710   3rd Qu.:0.0000           3rd Qu.: 2.053
##  Max.   :28.030   Max.   :4.9421           Max.   :21.920
##  ModeratelyActiveDistance LightActiveDistance SedentaryActiveDistance
##  Min.   :0.0000           Min.   : 0.000      Min.   :0.000000
##  1st Qu.:0.0000           1st Qu.: 1.945      1st Qu.:0.000000
##  Median :0.2400           Median : 3.365      Median :0.000000
##  Mean   :0.5675           Mean   : 3.341      Mean   :0.001606
##  3rd Qu.:0.8000           3rd Qu.: 4.782      3rd Qu.:0.000000
##  Max.   :6.4800           Max.   :10.710      Max.   :0.110000
##  VeryActiveMinutes FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes
##  Min.   :  0.00    Min.   :  0.00      Min.   :  0.0        Min.   :   0.0
##  1st Qu.:  0.00    1st Qu.:  0.00      1st Qu.:127.0        1st Qu.: 729.8
##  Median :  4.00    Median :  6.00      Median :199.0        Median :1057.5
##  Mean   : 21.16    Mean   : 13.56      Mean   :192.8        Mean   : 991.2
##  3rd Qu.: 32.00    3rd Qu.: 19.00      3rd Qu.:264.0        3rd Qu.:1229.5
##  Max.   :210.00    Max.   :143.00      Max.   :518.0        Max.   :1440.0
##     Calories
##  Min.   :   0
##  1st Qu.:1828
##  Median :2134
##  Mean   :2304
##  3rd Qu.:2793
##  Max.   :4900
```

```r
summary(daily_steps)
```

```
##        Id             ActivityDay          StepTotal
##  Min.   :1.504e+09   Length:940         Min.   :    0
##  1st Qu.:2.320e+09   Class :character   1st Qu.: 3790
##  Median :4.445e+09   Mode  :character   Median : 7406
##  Mean   :4.855e+09                      Mean   : 7638
##  3rd Qu.:6.962e+09                      3rd Qu.:10727
##  Max.   :8.878e+09                      Max.   :36019
```

```r
summary(heart_rate)
```

```
##        Id                 Time               Value
```

```
##  Min.   :2.022e+09   Length:2483658     Min.   : 36.00
##  1st Qu.:4.388e+09   Class :character   1st Qu.: 63.00
##  Median :5.554e+09   Mode  :character   Median : 73.00
##  Mean   :5.514e+09                      Mean   : 77.33
##  3rd Qu.:6.962e+09                      3rd Qu.: 88.00
##  Max.   :8.878e+09                      Max.   :203.00
```

```
glimpse(daily_activity)
```

```
## Rows: 940
## Columns: 15
## $ Id                       <dbl> 1503960366, 1503960366, 1503960366, 150396036~
## $ ActivityDate             <chr> "4/12/2016", "4/13/2016", "4/14/2016", "4/15/~
## $ TotalSteps               <dbl> 13162, 10735, 10460, 9762, 12669, 9705, 13019~
## $ TotalDistance            <dbl> 8.50, 6.97, 6.74, 6.28, 8.16, 6.48, 8.59, 9.8~
## $ TrackerDistance          <dbl> 8.50, 6.97, 6.74, 6.28, 8.16, 6.48, 8.59, 9.8~
## $ LoggedActivitiesDistance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ VeryActiveDistance       <dbl> 1.88, 1.57, 2.44, 2.14, 2.71, 3.19, 3.25, 3.5~
## $ ModeratelyActiveDistance <dbl> 0.55, 0.69, 0.40, 1.26, 0.41, 0.78, 0.64, 1.3~
## $ LightActiveDistance      <dbl> 6.06, 4.71, 3.91, 2.83, 5.04, 2.51, 4.71, 5.0~
## $ SedentaryActiveDistance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ VeryActiveMinutes        <dbl> 25, 21, 30, 29, 36, 38, 42, 50, 28, 19, 66, 4~
## $ FairlyActiveMinutes      <dbl> 13, 19, 11, 34, 10, 20, 16, 31, 12, 8, 27, 21~
## $ LightlyActiveMinutes     <dbl> 328, 217, 181, 209, 221, 164, 233, 264, 205, ~
## $ SedentaryMinutes         <dbl> 728, 776, 1218, 726, 773, 539, 1149, 775, 818~
## $ Calories                 <dbl> 1985, 1797, 1776, 1745, 1863, 1728, 1921, 203~
```

```
glimpse(daily_steps)
```

```
## Rows: 940
## Columns: 3
## $ Id          <dbl> 1503960366, 1503960366, 1503960366, 1503960366, 1503960366~
## $ ActivityDay <chr> "4/12/2016", "4/13/2016", "4/14/2016", "4/15/2016", "4/16/~
## $ StepTotal   <dbl> 13162, 10735, 10460, 9762, 12669, 9705, 13019, 15506, 1054~
```

```
glimpse(heart_rate)
```

```
## Rows: 2,483,658
## Columns: 3
## $ Id    <dbl> 2022484408, 2022484408, 2022484408, 2022484408, 2022484408, 2022~
## $ Time  <chr> "4/12/2016 7:21:00 AM", "4/12/2016 7:21:05 AM", "4/12/2016 7:21:~
## $ Value <dbl> 97, 102, 105, 103, 101, 95, 91, 93, 94, 93, 92, 89, 83, 61, 60, ~
```

Here, I will be changing the datatype of the Activity Date column into Date type.

```
daily_activity<-mutate(daily_activity,ActivityDate=as.Date(ActivityDate,"%m/%d/%Y"))
head(daily_activity)
```

```
## # A tibble: 6 x 15
##         Id ActivityDate TotalSteps TotalDistance TrackerDistance LoggedActivitie~
##      <dbl> <date>            <dbl>         <dbl>           <dbl>            <dbl>
```

```
## 1  1.50e9 2016-04-12        13162          8.5             8.5                   0
## 2  1.50e9 2016-04-13        10735          6.97            6.97                  0
## 3  1.50e9 2016-04-14        10460          6.74            6.74                  0
## 4  1.50e9 2016-04-15         9762          6.28            6.28                  0
## 5  1.50e9 2016-04-16        12669          8.16            8.16                  0
## 6  1.50e9 2016-04-17         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>
```

```
tail(daily_activity)
```

```
## # A tibble: 6 x 15
##         Id ActivityDate TotalSteps TotalDistance TrackerDistance LoggedActivitie~
##      <dbl> <date>            <dbl>         <dbl>           <dbl>            <dbl>
## 1  8.88e9 2016-05-07        12332          8.13            8.13                  0
## 2  8.88e9 2016-05-08        10686          8.11            8.11                  0
## 3  8.88e9 2016-05-09        20226         18.2            18.2                   0
## 4  8.88e9 2016-05-10        10733          8.15            8.15                  0
## 5  8.88e9 2016-05-11        21420         19.6            19.6                   0
## 6  8.88e9 2016-05-12         8064          6.12            6.12                  0
## # ... with 9 more variables: VeryActiveDistance <dbl>,
## #   ModeratelyActiveDistance <dbl>, LightActiveDistance <dbl>,
## #   SedentaryActiveDistance <dbl>, VeryActiveMinutes <dbl>,
## #   FairlyActiveMinutes <dbl>, LightlyActiveMinutes <dbl>,
## #   SedentaryMinutes <dbl>, Calories <dbl>
```

I wanted to know the number of customers whose data is present and used the unique function to find all the unique Id's. The output shows that the data consists of 33 customer data.

```
unique(daily_activity[c("Id")])
```

```
## # A tibble: 33 x 1
##            Id
##         <dbl>
##  1 1503960366
##  2 1624580081
##  3 1644430081
##  4 1844505072
##  5 1927972279
##  6 2022484408
##  7 2026352035
##  8 2320127002
##  9 2347167796
## 10 2873212765
## # ... with 23 more rows
```

I have tried to analyse of days of the week make any difference to the number of steps taken or the distance ceovered by the customers and hence added another column describing the Day of the week for each observation.

```
daily_activity$day_of_week<-format(as.Date(daily_activity$ActivityDate),"%A")
str(daily_activity)
```

```
## spec_tbl_df [940 x 16] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Id                      : num [1:940] 1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityDate            : Date[1:940], format: "2016-04-12" "2016-04-13" ...
##  $ TotalSteps              : num [1:940] 13162 10735 10460 9762 12669 ...
##  $ TotalDistance           : num [1:940] 8.5 6.97 6.74 6.28 8.16 ...
##  $ TrackerDistance         : num [1:940] 8.5 6.97 6.74 6.28 8.16 ...
##  $ LoggedActivitiesDistance: num [1:940] 0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveDistance      : num [1:940] 1.88 1.57 2.44 2.14 2.71 ...
##  $ ModeratelyActiveDistance: num [1:940] 0.55 0.69 0.4 1.26 0.41 ...
##  $ LightActiveDistance     : num [1:940] 6.06 4.71 3.91 2.83 5.04 ...
##  $ SedentaryActiveDistance : num [1:940] 0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveMinutes       : num [1:940] 25 21 30 29 36 38 42 50 28 19 ...
##  $ FairlyActiveMinutes     : num [1:940] 13 19 11 34 10 20 16 31 12 8 ...
##  $ LightlyActiveMinutes    : num [1:940] 328 217 181 209 221 164 233 264 205 211 ...
##  $ SedentaryMinutes        : num [1:940] 728 776 1218 726 773 ...
##  $ Calories                : num [1:940] 1985 1797 1776 1745 1863 ...
##  $ day_of_week             : chr [1:940] "Tuesday" "Wednesday" "Thursday" "Friday" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Id = col_double(),
##   ..   ActivityDate = col_character(),
##   ..   TotalSteps = col_double(),
##   ..   TotalDistance = col_double(),
##   ..   TrackerDistance = col_double(),
##   ..   LoggedActivitiesDistance = col_double(),
##   ..   VeryActiveDistance = col_double(),
##   ..   ModeratelyActiveDistance = col_double(),
##   ..   LightActiveDistance = col_double(),
##   ..   SedentaryActiveDistance = col_double(),
##   ..   VeryActiveMinutes = col_double(),
##   ..   FairlyActiveMinutes = col_double(),
##   ..   LightlyActiveMinutes = col_double(),
##   ..   SedentaryMinutes = col_double(),
##   ..   Calories = col_double()
##   .. )
```

We want the Days of the week to be ordered well so the data will be easier to understand.

```
daily_activity$day_of_week<-ordered(daily_activity$day_of_week,level=c("Sunday","Monday","Tuesday","Wedn
```

I went ahead to summarise the average steps and average distance covered by the customers based on the week day.

```
daily_activity%>%
  group_by(day_of_week)%>%
  summarise(average_steps=mean(TotalSteps),average_distance=mean(TotalDistance))
```

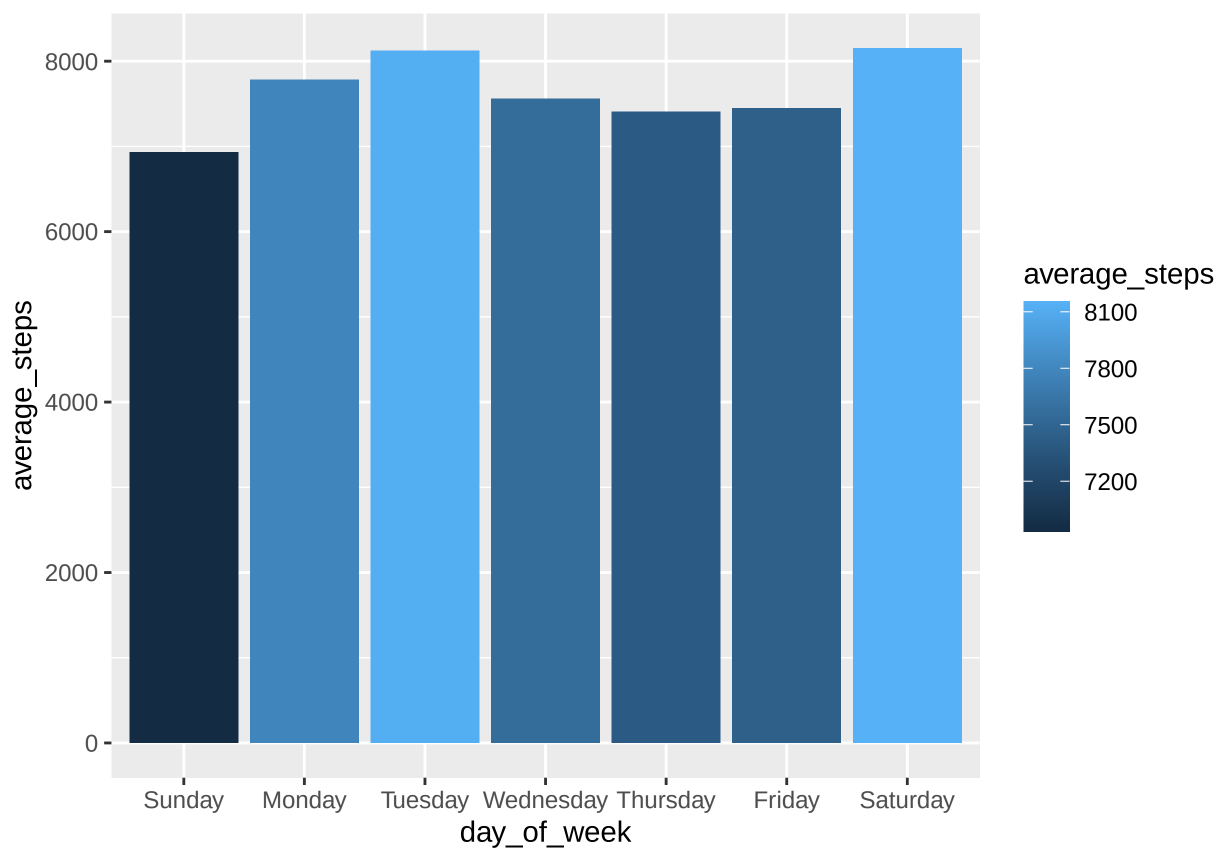
```
## # A tibble: 7 x 3
##   day_of_week average_steps average_distance
```

```
##   <ord>                <dbl>           <dbl>
## 1 Sunday               6933.            5.03
## 2 Monday               7781.            5.55
## 3 Tuesday              8125.            5.83
## 4 Wednesday            7559.            5.49
## 5 Thursday             7406.            5.31
## 6 Friday               7448.            5.31
## 7 Saturday             8153.            5.85
```

As we can see in the output, the distance covered by the customers is consistent throughout but the steps covered varies.

The graph plotted below helps better understand the trend of steps over the week days

```
daily_activity%>%
  group_by(day_of_week)%>%
  summarise(average_steps=mean(TotalSteps),average_distance=mean(TotalDistance))%>%
  ggplot(aes(x=day_of_week,y=average_steps,fill=average_steps))+geom_col(position="dodge")
```



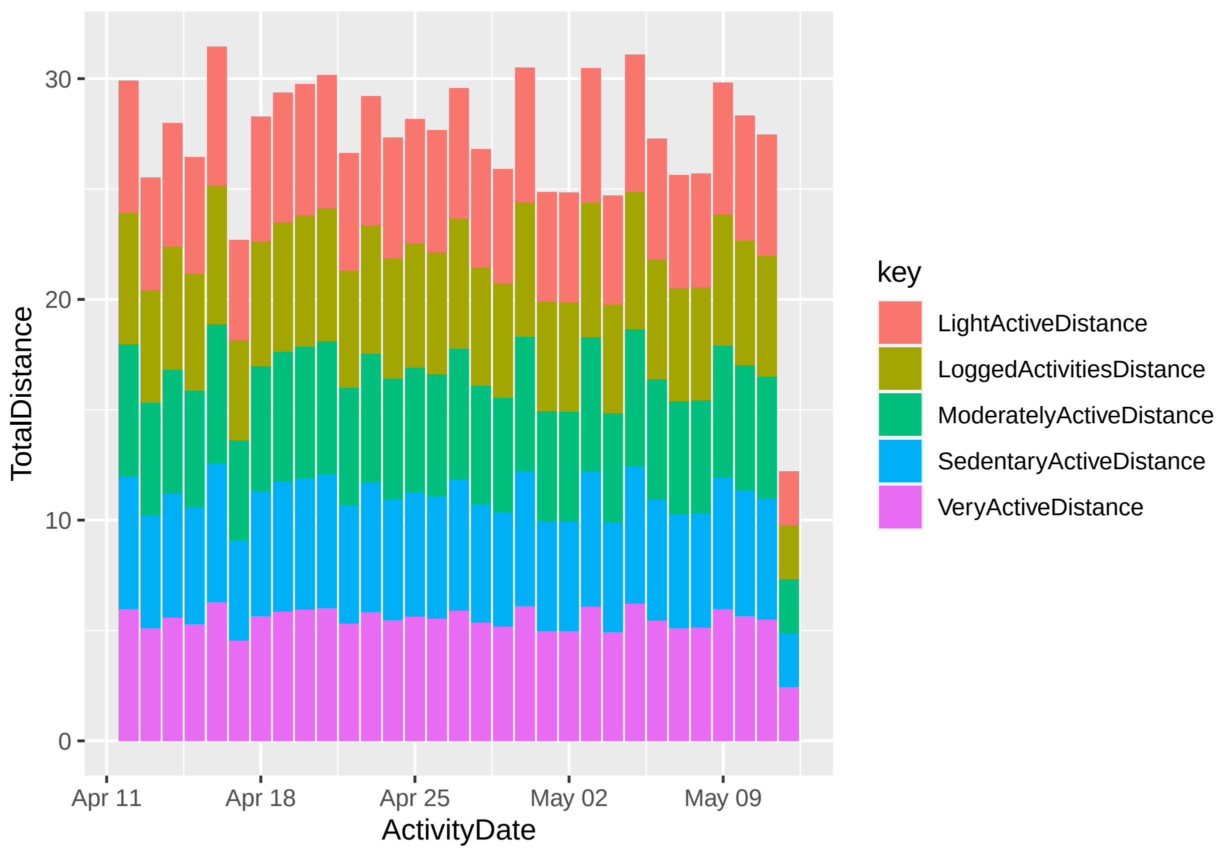
The customers seem to be highly active over tuesdays and Saturdays. I next tried to summarize , out of the total distance covered by the customer, how much of the distance covered has been effective.

```
distance_data<-daily_activity%>%
  group_by(ActivityDate)%>%
  summarise(TotalDistance=mean(TotalDistance),LoggedActivitiesDistance=mean(LoggedActivitiesDistance),V
distance_data
```

```
## # A tibble: 31 x 7
##    ActivityDate TotalDistance LoggedActivitie~ VeryActiveDista~ ModeratelyActiv~
##    <date>               <dbl>            <dbl>            <dbl>            <dbl>
##  1 2016-04-12            5.98            0.216             1.83            0.346
##  2 2016-04-13            5.10            0.210             1.33            0.420
##  3 2016-04-14            5.60            0.168             1.51            0.510
##  4 2016-04-15            5.29            0                 1.06            0.404
##  5 2016-04-16            6.29            0                 1.99            0.709
##  6 2016-04-17            4.54            0                 1.15            0.497
##  7 2016-04-18            5.66            0.219             1.67            0.696
##  8 2016-04-19            5.87            0.225             1.88            0.519
##  9 2016-04-20            5.95            0.219             1.86            0.633
## 10 2016-04-21            6.03            0.198             1.92            0.622
## # ... with 21 more rows, and 2 more variables: LightActiveDistance <dbl>,
## #   SedentaryActiveDistance <dbl>
```

The graph below shows the average active distances over the period of 30 days by the 33 customers and the variations of active distances.

```
distance_data%>%
  gather(key,value,-c(ActivityDate,TotalDistance))%>%
  ggplot(aes(fill=key,y=TotalDistance,x=ActivityDate))+
  geom_col()
```



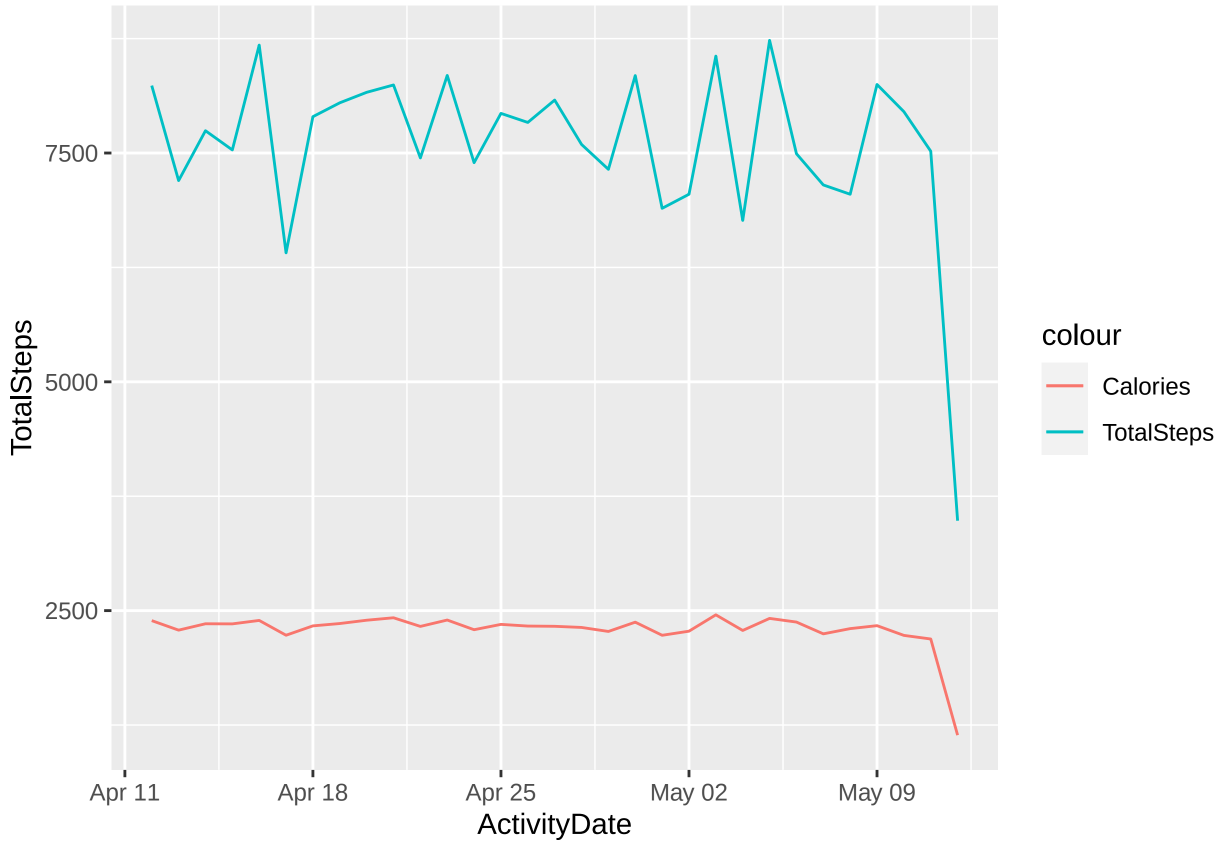
In order to analyze any correlation between the steps and calories burnt , the following summery was helpful.

```
daily_activity%>%
  group_by(Id)%>%
  summarise(TotalSteps=mean(TotalSteps),Calories=mean(Calories))
```

```
## # A tibble: 33 x 3
##            Id TotalSteps Calories
##         <dbl>      <dbl>    <dbl>
##  1 1503960366     12117.    1816.
##  2 1624580081      5744.    1483.
##  3 1644430081      7283.    2811.
##  4 1844505072      2580.    1573.
##  5 1927972279       916.    2173.
##  6 2022484408     11371.    2510.
##  7 2026352035      5567.    1541.
##  8 2320127002      4717.    1724.
##  9 2347167796      9520.    2043.
## 10 2873212765      7556.    1917.
## # ... with 23 more rows
```

The graph below shows the correlation between the two variables and they seem to be related.

```
daily_activity%>%
  group_by(ActivityDate)%>%
  summarise(TotalSteps=mean(TotalSteps),Calories=mean(Calories))%>%
  ggplot()+
  geom_line(aes(y=TotalSteps,x=ActivityDate,color="TotalSteps"))+
  geom_line(aes(y=Calories,x=ActivityDate,color="Calories"))
```



I tried to extract the two IDs of customers with the most minimum and maximum records within the data.

```
max_min_values<-daily_activity%>%
  group_by(Id)%>%
  summarise(average_steps=mean(TotalSteps),average_distance=mean(TotalDistance))%>%
  filter(average_steps==max(average_steps)| average_steps==min(average_steps) | average_distance==max(a
max_min_values
```

```
## # A tibble: 2 x 3
##           Id average_steps average_distance
##        <dbl>         <dbl>            <dbl>
## 1 1927972279          916.            0.635
## 2 8877689391        16040.           13.2
```

The week data of both the Ids is now retrieved to see their weekly walk habits.

```
daily_activity%>%
  right_join(max_min_values,by="Id")%>%
  group_by(Id,day_of_week)%>%
  summarise(average_steps=mean(TotalSteps),average_distance=mean(TotalDistance))
```

```
## `summarise()` has grouped output by 'Id'. You can override using the `.groups` argument.
```

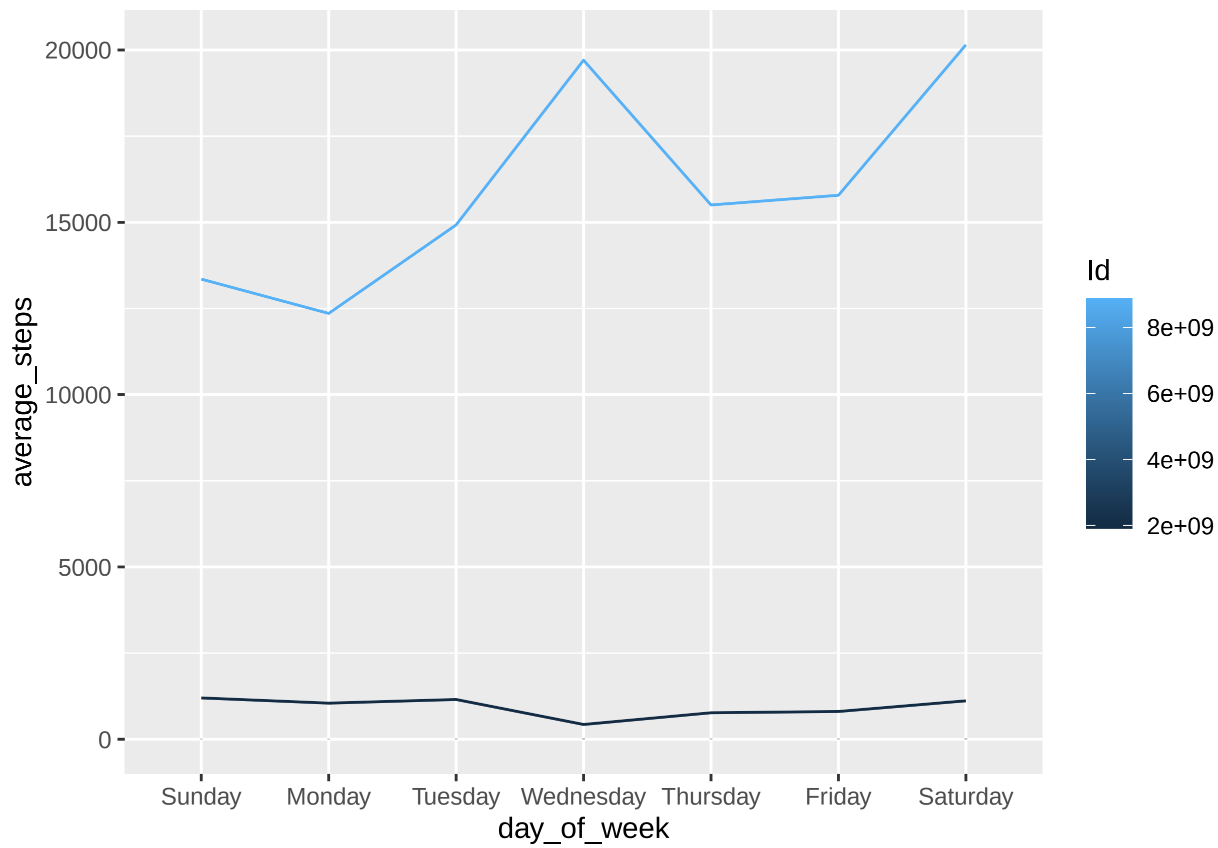
```
## # A tibble: 14 x 4
## # Groups:   Id [2]
```

```
##            Id day_of_week average_steps average_distance
##         <dbl> <ord>               <dbl>            <dbl>
##  1 1927972279 Sunday              1198.            0.830
##  2 1927972279 Monday              1046.            0.725
##  3 1927972279 Tuesday             1153             0.798
##  4 1927972279 Wednesday            428.            0.298
##  5 1927972279 Thursday             768.            0.532
##  6 1927972279 Friday               805             0.558
##  7 1927972279 Saturday            1114.            0.770
##  8 8877689391 Sunday             13352            10.3
##  9 8877689391 Monday             12356.            9.82
## 10 8877689391 Tuesday            14925.           12.5
## 11 8877689391 Wednesday          19705.           16.8
## 12 8877689391 Thursday           15503            12.9
## 13 8877689391 Friday             15785            12.6
## 14 8877689391 Saturday           20151.           16.9
```

The graph below shows the two steps trend of the two Ids throughthe weekdays. We can see that while the number of steps covered by the lower ID is less, it is consistent throught the week. The other Id however seems to be highly motivated mid-week and saturdays.

```
daily_activity$day_of_week<-ordered(daily_activity$day_of_week,level=c("Sunday","Monday","Tuesday","Wedn
daily_activity%>%
  right_join(max_min_values,by="Id")%>%
  group_by(Id,day_of_week)%>%
  summarise(average_steps=mean(TotalSteps),average_distance=mean(TotalDistance))%>%
  ggplot()+geom_line(aes(x=day_of_week,y=average_steps,group=Id,color=Id))+
  geom_line(aes(x=day_of_week,y=average_distance,color=Id))
```

```
## `summarise()` has grouped output by 'Id'. You can override using the `.groups` argument.
```



I have used the heart beat data to see if the highly active customer has linear effect of steps on the heart beat rate.

```
head(heart_rate)
```

```
## # A tibble: 6 x 3
##           Id Time                  Value
##        <dbl> <chr>                 <dbl>
## 1 2022484408 4/12/2016 7:21:00 AM     97
## 2 2022484408 4/12/2016 7:21:05 AM    102
## 3 2022484408 4/12/2016 7:21:10 AM    105
## 4 2022484408 4/12/2016 7:21:20 AM    103
## 5 2022484408 4/12/2016 7:21:25 AM    101
## 6 2022484408 4/12/2016 7:22:05 AM     95
```

```
heart_rate$date_time=mdy_hms(heart_rate$Time)
heart_rate$hdate=ymd(date(heart_rate$date_time))
```

```
tail(heart_rate)
```

```
## # A tibble: 6 x 5
##           Id Time                  Value date_time           hdate
##        <dbl> <chr>                 <dbl> <dttm>              <date>
## 1 8877689391 5/12/2016 2:43:38 PM     58 2016-05-12 14:43:38 2016-05-12
## 2 8877689391 5/12/2016 2:43:53 PM     57 2016-05-12 14:43:53 2016-05-12
```

```
## 3 8877689391 5/12/2016 2:43:58 PM    56 2016-05-12 14:43:58 2016-05-12
## 4 8877689391 5/12/2016 2:44:03 PM    55 2016-05-12 14:44:03 2016-05-12
## 5 8877689391 5/12/2016 2:44:18 PM    55 2016-05-12 14:44:18 2016-05-12
## 6 8877689391 5/12/2016 2:44:28 PM    56 2016-05-12 14:44:28 2016-05-12
```

```
max_heart_beats<-heart_rate%>%
  group_by(Id,hdate)%>%
  right_join(max_min_values,by="Id")%>%
  summarise(average_heartbeat_per_day=mean(Value))
```

```
## `summarise()` has grouped output by 'Id'. You can override using the `.groups` argument.
```

```
drop_na(max_heart_beats,hdate)
```

```
## # A tibble: 31 x 3
## # Groups:   Id [1]
##            Id hdate      average_heartbeat_per_day
##         <dbl> <date>                         <dbl>
##  1 8877689391 2016-04-12                      86.5
##  2 8877689391 2016-04-13                      84.7
##  3 8877689391 2016-04-14                      85.3
##  4 8877689391 2016-04-15                      91.9
##  5 8877689391 2016-04-16                      92.1
##  6 8877689391 2016-04-17                      87.9
##  7 8877689391 2016-04-18                      66.0
##  8 8877689391 2016-04-19                      86.6
##  9 8877689391 2016-04-20                      85.5
## 10 8877689391 2016-04-21                      87.1
## # ... with 21 more rows
```

```
drop_na(max_heart_beats,average_heartbeat_per_day)
```

```
## # A tibble: 31 x 3
## # Groups:   Id [1]
##            Id hdate      average_heartbeat_per_day
##         <dbl> <date>                         <dbl>
##  1 8877689391 2016-04-12                      86.5
##  2 8877689391 2016-04-13                      84.7
##  3 8877689391 2016-04-14                      85.3
##  4 8877689391 2016-04-15                      91.9
##  5 8877689391 2016-04-16                      92.1
##  6 8877689391 2016-04-17                      87.9
##  7 8877689391 2016-04-18                      66.0
##  8 8877689391 2016-04-19                      86.6
##  9 8877689391 2016-04-20                      85.5
## 10 8877689391 2016-04-21                      87.1
## # ... with 21 more rows
```

```
glimpse(max_heart_beats)
```

```
## Rows: 32
```

```
## Columns: 3
## Groups: Id [2]
## $ Id                          <dbl> 1927972279, 8877689391, 8877689391, 88776893~
## $ hdate                       <date> NA, 2016-04-12, 2016-04-13, 2016-04-14, 201~
## $ average_heartbeat_per_day <dbl> NA, 86.54771, 84.74373, 85.32065, 91.93236, ~
```

The daily_steps data frame also consisted of character data for the dates column. The data type is changed for easy access and manipulation of the data

```
daily_steps<-mutate(daily_steps,ActivityDay=as.Date(ActivityDay,"%m/%d/%Y"))
glimpse(daily_steps)
```

```
## Rows: 940
## Columns: 3
## $ Id          <dbl> 1503960366, 1503960366, 1503960366, 1503960366, 1503960366~
## $ ActivityDay <date> 2016-04-12, 2016-04-13, 2016-04-14, 2016-04-15, 2016-04-1~
## $ StepTotal   <dbl> 13162, 10735, 10460, 9762, 12669, 9705, 13019, 15506, 1054~
```

The average steps of each day for the period os one month is calculated for the particular customer.

```
steps_data<-daily_steps%>%
  group_by(ActivityDay,Id)%>%
  right_join(max_heart_beats,by="Id")%>%
  summarise(steps_per_day=mean(StepTotal))
```

```
## `summarise()` has grouped output by 'ActivityDay'. You can override using the `.groups` argument.
```

```
steps_data
```

```
## # A tibble: 62 x 3
## # Groups:   ActivityDay [31]
##    ActivityDay         Id steps_per_day
##    <date>           <dbl>         <dbl>
##  1 2016-04-12  1927972279           678
##  2 2016-04-12  8877689391         23186
##  3 2016-04-13  1927972279           356
##  4 2016-04-13  8877689391         15337
##  5 2016-04-14  1927972279          2163
##  6 2016-04-14  8877689391         21129
##  7 2016-04-15  1927972279           980
##  8 2016-04-15  8877689391         13422
##  9 2016-04-16  1927972279             0
## 10 2016-04-16  8877689391         29326
## # ... with 52 more rows
```

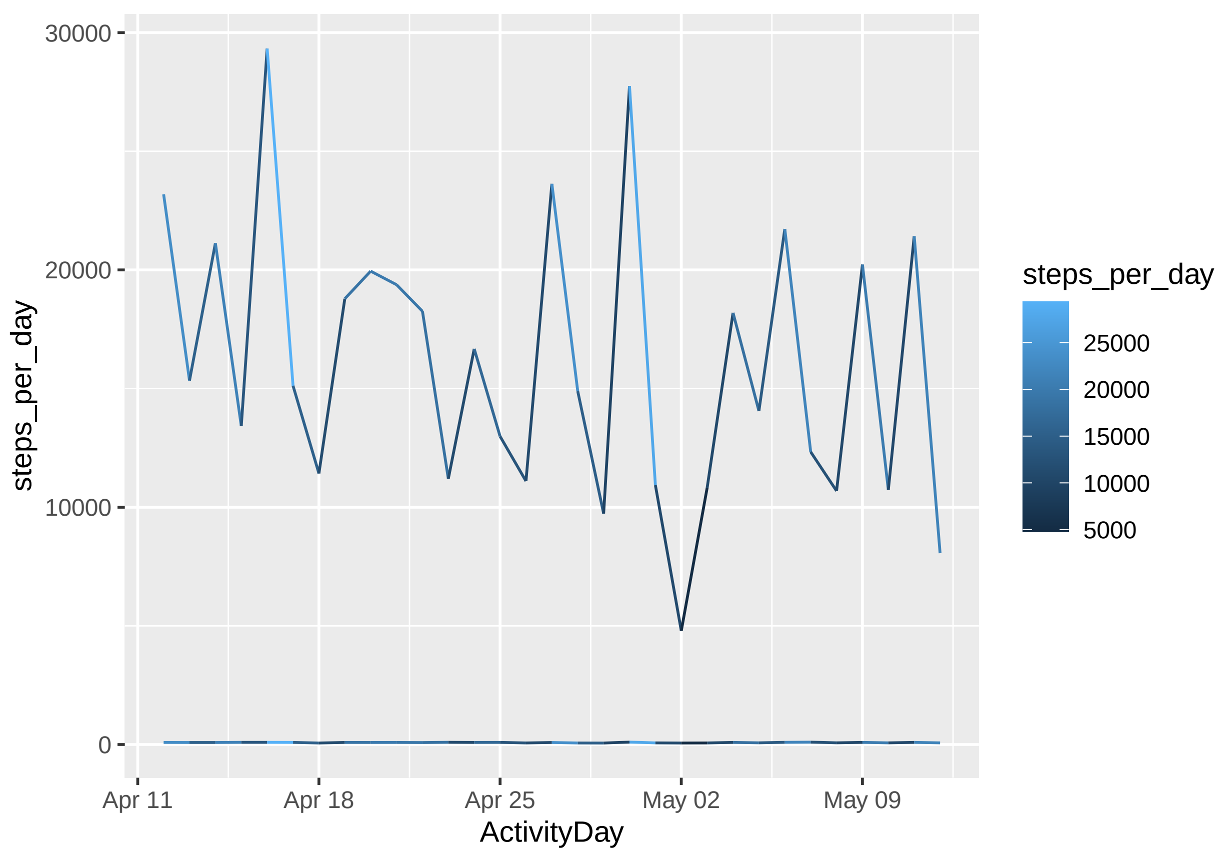
The average heartbeat of the same customer for each day is calculated and combined with steps data to obtain a summary

```
steps_heart<-steps_data%>%
  left_join(max_heart_beats,by='Id')%>%
  filter(ActivityDay==hdate)%>%
  select(-c(hdate,Id))
steps_heart
```

```
## # A tibble: 31 x 3
## # Groups:   ActivityDay [31]
##    ActivityDay steps_per_day average_heartbeat_per_day
##    <date>              <dbl>                     <dbl>
##  1 2016-04-12          23186                      86.5
##  2 2016-04-13          15337                      84.7
##  3 2016-04-14          21129                      85.3
##  4 2016-04-15          13422                      91.9
##  5 2016-04-16          29326                      92.1
##  6 2016-04-17          15118                      87.9
##  7 2016-04-18          11423                      66.0
##  8 2016-04-19          18785                      86.6
##  9 2016-04-20          19948                      85.5
## 10 2016-04-21          19377                      87.1
## # ... with 21 more rows
```

We can observe from the graph that the number of steps and the heart beat are not correlation for this particular customer

```
steps_heart%>%
  ggplot()+
  geom_line(aes(x=ActivityDay,y=steps_per_day,color=steps_per_day))+
  geom_line(aes(x=ActivityDay,y=average_heartbeat_per_day,color=steps_per_day))
```



The large data set increases the scope of analysis and in-depth analysis will help Bellabeat of find trends in data that could help them to improve their business.