# Case Study: How Can a Wellness Technology Company Play It Smart?

**prepared by : Yasin MASLAK**

*date : 12.05.2024*

---

**Bellabeat Wellness Technology**

Welcome to the Bellabeat data analysis case study! In this case study, you will perform many real-world tasks of a junior data analyst. You will imagine you are working for Bellabeat, a high-tech manufacturer of health-focused products for women, and meet different characters and team members. In order to answer the key business questions, you will follow the steps of the data analysis process: ask, prepare, process, analyze, share, and act. Along the way, the Case Study Roadmap tables — including guiding questions and key tasks — will help you stay on the right path.

**Ask Problem:**

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

The data isn't maximized in providing better business decisions for the company

Business Task: 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, make a like high-level recommendations for how these trends can inform Bellabeat marketing strategy.

**Guiding questions**

● How should you organize your data to perform analysis on it?
● Has your data been properly formatted?
● What surprises did you discover in the data?
● What trends or relationships did you find in the data?
● How will these insights help answer your business questions?

**Initial Thoughts:**

The demographic and age may be a factor in predicting the people who are most likely to buy. Key Stakeholders: Urska Srsen, Sando Mur, Marketing Analytics Team

Possible data & metrics: usage, age distribution, smartphone devices

FitBit Fitness Tracker Data : [https://www.kaggle.com/datasets/arashnic/fitbit](https://www.kaggle.com/datasets/arashnic/fitbit)

Same Data Anayze with Python : [https://www.kaggle.com/code/mahmoudosama10/fitbit-fitness-tracker-analysis](https://www.kaggle.com/code/mahmoudosama10/fitbit-fitness-tracker-analysis)

[https://www.kaggle.com/code/abhishekp297/google-data-analytics-case-study-2-python](https://www.kaggle.com/code/abhishekp297/google-data-analytics-case-study-2-python)

```
#Importing Necessary Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
#reading the dataset
daily_activity = pd.read_csv('/content/drive/MyDrive/ColabNotebooks/Case_Study_2/dailyActivity_merged.csv')
daily_activity
```

| | Id | ActivityDate | TotalSteps | TotalDistance | TrackerDistance | LoggedAct |
|---|---|---|---|---|---|---|
| **0** | 1503960366 | 4/12/2016 | 13162 | 8.500000 | 8.500000 | |
| **1** | 1503960366 | 4/13/2016 | 10735 | 6.970000 | 6.970000 | |
| **2** | 1503960366 | 4/14/2016 | 10460 | 6.740000 | 6.740000 | |
| **3** | 1503960366 | 4/15/2016 | 9762 | 6.280000 | 6.280000 | |
| **4** | 1503960366 | 4/16/2016 | 12669 | 8.160000 | 8.160000 | |
| **...** | ... | ... | ... | ... | ... | |
| **935** | 8877689391 | 5/8/2016 | 10686 | 8.110000 | 8.110000 | |
| **936** | 8877689391 | 5/9/2016 | 20226 | 18.250000 | 18.250000 | |
| **937** | 8877689391 | 5/10/2016 | 10733 | 8.150000 | 8.150000 | |
| **938** | 8877689391 | 5/11/2016 | 21420 | 19.559999 | 19.559999 | |
| **939** | 8877689391 | 5/12/2016 | 8064 | 6.120000 | 6.120000 | |

940 rows × 15 columns

Next steps: Generate code with `daily_activity` View recommended plots

## these datas has been reeding with daily_activity

- daily_Calories
- dailyIntensities dailyIntensities = pd.read_csv('/content/drive/MyDrive/ColabNotebooks/Case_Study_2/dailyIntensities_merged.csv')

```
#Before de analyzes I checked each table is part of dataset for maximum information
weightLogInfo = pd.read_csv('/content/drive/MyDrive/ColabNotebooks/Case_Study_2/weightLogInfo_merged.csv')
weightLogInfo #this dataset has been not reeding with daily_activity
```

| | Id | Date | WeightKg | WeightPounds | Fat | BMI | IsManualReport |
|---|---|---|---|---|---|---|---|
| **0** | 1503960366 | 5/2/2016 11:59:59 PM | 52.599998 | 115.963147 | 22.0 | 22.650000 | True |
| **1** | 1503960366 | 5/3/2016 11:59:59 PM | 52.599998 | 115.963147 | NaN | 22.650000 | True |
| **2** | 1927972279 | 4/13/2016 1:08:52 AM | 133.500000 | 294.317120 | NaN | 47.540001 | False |
| **3** | 2873212765 | 4/21/2016 11:59:59 PM | 56.700001 | 125.002104 | NaN | 21.450001 | True |
| **4** | 2873212765 | 5/12/2016 11:59:59 PM | 57.299999 | 126.324875 | NaN | 21.690001 | True |
| **...** | ... | ... | ... | ... | ... | ... | ... |

Next steps: Generate code with `weightLogInfo` View recommended plots

```
#a brief informaton our first dataset
daily_activity.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 940 entries, 0 to 939
Data columns (total 15 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Id                        940 non-null    int64  
 1   ActivityDate              940 non-null    object 
 2   TotalSteps                940 non-null    int64  
 3   TotalDistance             940 non-null    float64
 4   TrackerDistance           940 non-null    float64
 5   LoggedActivitiesDistance  940 non-null    float64
 6   VeryActiveDistance        940 non-null    float64
 7   ModeratelyActiveDistance  940 non-null    float64
 8   LightActiveDistance       940 non-null    float64
 9   SedentaryActiveDistance   940 non-null    float64
 10  VeryActiveMinutes         940 non-null    int64  
```

```
 11  FairlyActiveMinutes       940 non-null    int64
 12  LightlyActiveMinutes      940 non-null    int64
 13  SedentaryMinutes          940 non-null    int64
 14  Calories                  940 non-null    int64
dtypes: float64(7), int64(7), object(1)
memory usage: 110.3+ KB
```

```
#e brief informaton our second dataset
weightLogInfo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 67 entries, 0 to 66
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Id              67 non-null     int64
 1   Date            67 non-null     object
 2   WeightKg        67 non-null     float64
 3   WeightPounds    67 non-null     float64
 4   Fat             2 non-null      float64
 5   BMI             67 non-null     float64
 6   IsManualReport  67 non-null     bool
 7   LogId           67 non-null     int64
dtypes: bool(1), float64(4), int64(2), object(1)
memory usage: 3.9+ KB
```

** Fist Step : Understanding Dataset**

In first step we focus on fist dataset because it can gives us to max knowledge second dataset may be can useful after analyze first one

```
daily_activity = daily_activity.copy()
daily_activity.head()
```

| | Id | ActivityDate | TotalSteps | TotalDistance | TrackerDistance | LoggedActiv |
|---|---|---|---|---|---|---|
| 0 | 1503960366 | 4/12/2016 | 13162 | 8.50 | 8.50 | |
| 1 | 1503960366 | 4/13/2016 | 10735 | 6.97 | 6.97 | |
| 2 | 1503960366 | 4/14/2016 | 10460 | 6.74 | 6.74 | |
| 3 | 1503960366 | 4/15/2016 | 9762 | 6.28 | 6.28 | |
| 4 | 1503960366 | 4/16/2016 | 12669 | 8.16 | 8.16 | |

Next steps: Generate code with daily_activity View recommended plots

```
daily_activity.columns
```

```
Index(['Id', 'ActivityDate', 'TotalSteps', 'TotalDistance', 'TrackerDistance',
       'LoggedActivitiesDistance', 'VeryActiveDistance',
       'ModeratelyActiveDistance', 'LightActiveDistance',
       'SedentaryActiveDistance', 'VeryActiveMinutes', 'FairlyActiveMinutes',
       'LightlyActiveMinutes', 'SedentaryMinutes', 'Calories'],
      dtype='object')
```

## Dataset Column names

*'Id' *'ActivityDate', *'TotalSteps', *'TotalDistance', *'TrackerDistance', *'LoggedActivitiesDistance', *'VeryActiveDistance', *'ModeratelyActiveDistance', *'LightActiveDistance', *'SedentaryActiveDistance',* VeryActiveMinutes', '*FairlyActiveMinutes',* LightlyActiveMinutes', '*SedentaryMinutes', *'Calories

```
daily_activity.describe().T
```

| | count | mean | std | min | 25% |
|---|---|---|---|---|---|
| **Id** | 940.0 | 4.855407e+09 | 2.424805e+09 | 1.503960e+09 | 2.320127e+09 |
| **TotalSteps** | 940.0 | 7.637911e+03 | 5.087151e+03 | 0.000000e+00 | 3.789750e+03 |
| **TotalDistance** | 940.0 | 5.489702e+00 | 3.924606e+00 | 0.000000e+00 | 2.620000e+00 |
| **TrackerDistance** | 940.0 | 5.475351e+00 | 3.907276e+00 | 0.000000e+00 | 2.620000e+00 |
| **LoggedActivitiesDistance** | 940.0 | 1.081709e-01 | 6.198965e-01 | 0.000000e+00 | 0.000000e+00 |
| **VeryActiveDistance** | 940.0 | 1.502681e+00 | 2.658941e+00 | 0.000000e+00 | 0.000000e+00 |
| **ModeratelyActiveDistance** | 940.0 | 5.675426e-01 | 8.835803e-01 | 0.000000e+00 | 0.000000e+00 |
| **LightActiveDistance** | 940.0 | 3.340819e+00 | 2.040655e+00 | 0.000000e+00 | 1.945000e+00 |
| **SedentaryActiveDistance** | 940.0 | 1.606383e-03 | 7.346176e-03 | 0.000000e+00 | 0.000000e+00 |
| **VeryActiveMinutes** | 940.0 | 2.116489e+01 | 3.284480e+01 | 0.000000e+00 | 0.000000e+00 |
| **FairlyActiveMinutes** | 940.0 | 1.356489e+01 | 1.998740e+01 | 0.000000e+00 | 0.000000e+00 |
| **LightlyActiveMinutes** | 940.0 | 1.928128e+02 | 1.091747e+02 | 0.000000e+00 | 1.270000e+02 |
| **SedentaryMinutes** | 940.0 | 9.912106e+02 | 3.012674e+02 | 0.000000e+00 | 7.297500e+02 |
| **Calories** | 940.0 | 2.303610e+03 | 7.181669e+02 | 0.000000e+00 | 1.828500e+03 |

```
#To check null datas
daily_activity.isnull().values.any() #False null data is not exist our dataset
```

```
False
```

```
#Variables types is important to correct informations during analyze
daily_activity.dtypes
```

```
Id                            int64
ActivityDate                 object
TotalSteps                    int64
TotalDistance               float64
TrackerDistance             float64
LoggedActivitiesDistance    float64
VeryActiveDistance          float64
ModeratelyActiveDistance    float64
LightActiveDistance         float64
SedentaryActiveDistance     float64
VeryActiveMinutes             int64
FairlyActiveMinutes           int64
LightlyActiveMinutes          int64
SedentaryMinutes              int64
Calories                      int64
dtype: object
```

## we here focus on totalsteps and minutes and it's relationship with calories burned.

### * we will remove the unnecessary columns

```
columns = ['Id', 'ActivityDate', 'TotalSteps', 'VeryActiveMinutes', 'FairlyActiveMinutes',
           'LightlyActiveMinutes', 'SedentaryMinutes', 'Calories']

daily_activity = daily_activity[columns]
daily_activity
```

| | Id | ActivityDate | TotalSteps | VeryActiveMinutes | FairlyActiveMinutes | L |
|---|---|---|---|---|---|---|
| **0** | 1503960366 | 4/12/2016 | 13162 | 25 | 13 | |
| **1** | 1503960366 | 4/13/2016 | 10735 | 21 | 19 | |
| **2** | 1503960366 | 4/14/2016 | 10460 | 30 | 11 | |
| **3** | 1503960366 | 4/15/2016 | 9762 | 29 | 34 | |
| **4** | 1503960366 | 4/16/2016 | 12669 | 36 | 10 | |
| **...** | ... | ... | ... | ... | ... | |
| **935** | 8877689391 | 5/8/2016 | 10686 | 17 | 4 | |
| **936** | 8877689391 | 5/9/2016 | 20226 | 73 | 19 | |
| **937** | 8877689391 | 5/10/2016 | 10733 | 18 | 11 | |
| **938** | 8877689391 | 5/11/2016 | 21420 | 88 | 12 | |
| **939** | 8877689391 | 5/12/2016 | 8064 | 23 | 1 | |

940 rows × 8 columns

Next steps: Generate code with `daily_activity` View recommended plots

```
daily_activity.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 940 entries, 0 to 939
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Id                    940 non-null    int64
 1   ActivityDate          940 non-null    object
 2   TotalSteps            940 non-null    int64
 3   VeryActiveMinutes     940 non-null    int64
 4   FairlyActiveMinutes   940 non-null    int64
 5   LightlyActiveMinutes  940 non-null    int64
 6   SedentaryMinutes      940 non-null    int64
 7   Calories              940 non-null    int64
dtypes: int64(7), object(1)
memory usage: 58.9+ KB
```

```
daily_activity['TotalMinutes'] = daily_activity['VeryActiveMinutes'] + daily_activity['FairlyActiveMinutes'] + daily_activity['LightlyA
daily_activity
```

| | Id | ActivityDate | TotalSteps | VeryActiveMinutes | FairlyActiveMinutes | L |
|---|---|---|---|---|---|---|
| **0** | 1503960366 | 4/12/2016 | 13162 | 25 | 13 | |
| **1** | 1503960366 | 4/13/2016 | 10735 | 21 | 19 | |
| **2** | 1503960366 | 4/14/2016 | 10460 | 30 | 11 | |
| **3** | 1503960366 | 4/15/2016 | 9762 | 29 | 34 | |
| **4** | 1503960366 | 4/16/2016 | 12669 | 36 | 10 | |
| **...** | ... | ... | ... | ... | ... | |
| **935** | 8877689391 | 5/8/2016 | 10686 | 17 | 4 | |
| **936** | 8877689391 | 5/9/2016 | 20226 | 73 | 19 | |
| **937** | 8877689391 | 5/10/2016 | 10733 | 18 | 11 | |
| **938** | 8877689391 | 5/11/2016 | 21420 | 88 | 12 | |
| **939** | 8877689391 | 5/12/2016 | 8064 | 23 | 1 | |

940 rows × 9 columns

Next steps: Generate code with `daily_activity` View recommended plots

```
daily_activity['TotalHours'] = round(daily_activity.TotalMinutes / 60)
daily_activity
```

| | Id | ActivityDate | TotalSteps | VeryActiveMinutes | FairlyActiveMinutes | L |
|---|---|---|---|---|---|---|
| **0** | 1503960366 | 4/12/2016 | 13162 | 25 | 13 | |
| **1** | 1503960366 | 4/13/2016 | 10735 | 21 | 19 | |
| **2** | 1503960366 | 4/14/2016 | 10460 | 30 | 11 | |
| **3** | 1503960366 | 4/15/2016 | 9762 | 29 | 34 | |
| **4** | 1503960366 | 4/16/2016 | 12669 | 36 | 10 | |
| **...** | ... | ... | ... | ... | ... | |
| **935** | 8877689391 | 5/8/2016 | 10686 | 17 | 4 | |
| **936** | 8877689391 | 5/9/2016 | 20226 | 73 | 19 | |
| **937** | 8877689391 | 5/10/2016 | 10733 | 18 | 11 | |
| **938** | 8877689391 | 5/11/2016 | 21420 | 88 | 12 | |
| **939** | 8877689391 | 5/12/2016 | 8064 | 23 | 1 | |

940 rows × 10 columns

Next steps: Generate code with daily_activity View recommended plots

```
#convert the ActivityDate column to datetime type not object
# Before analyze is so important because wrong datetype gives wrong result in each step
daily_activity.ActivityDate = pd.to_datetime(daily_activity.ActivityDate)
daily_activity.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 940 entries, 0 to 939
Data columns (total 10 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Id                    940 non-null    int64
 1   ActivityDate          940 non-null    datetime64[ns]
 2   TotalSteps            940 non-null    int64
 3   VeryActiveMinutes     940 non-null    int64
 4   FairlyActiveMinutes   940 non-null    int64
 5   LightlyActiveMinutes  940 non-null    int64
 6   SedentaryMinutes      940 non-null    int64
 7   Calories              940 non-null    int64
 8   TotalMinutes          940 non-null    int64
 9   TotalHours            940 non-null    float64
dtypes: datetime64[ns](1), float64(1), int64(8)
memory usage: 73.6 KB
```

```
# We catch days of Week and add as a column
daily_activity['DayOfWeek'] = daily_activity.ActivityDate.dt.day_name()
daily_activity
```

| | Id | ActivityDate | TotalSteps | VeryActiveMinutes | FairlyActiveMinutes | L |
|---|---|---|---|---|---|---|
| **0** | 1503960366 | 2016-04-12 | 13162 | 25 | 13 | |
| **1** | 1503960366 | 2016-04-13 | 10735 | 21 | 19 | |
| **2** | 1503960366 | 2016-04-14 | 10460 | 30 | 11 | |
| **3** | 1503960366 | 2016-04-15 | 9762 | 29 | 34 | |
| **4** | 1503960366 | 2016-04-16 | 12669 | 36 | 10 | |
| **...** | ... | ... | ... | ... | ... | |
| **935** | 8877689391 | 2016-05-08 | 10686 | 17 | 4 | |
| **936** | 8877689391 | 2016-05-09 | 20226 | 73 | 19 | |
| **937** | 8877689391 | 2016-05-10 | 10733 | 18 | 11 | |
| **938** | 8877689391 | 2016-05-11 | 21420 | 88 | 12 | |
| **939** | 8877689391 | 2016-05-12 | 8064 | 23 | 1 | |

940 rows × 11 columns

Next steps: Generate code with daily_activity View recommended plots

Kodlamaya başlayın veya yapay zeka ile kod oluşturun.

```
days_ordered= ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']

daily_activity['DayOfWeek'] = pd.Categorical(daily_activity['DayOfWeek'], categories=days_ordered, ordered=True)
daily_activity = daily_activity.sort_values('DayOfWeek')
daily_activity
```

| | Id | ActivityDate | TotalSteps | VeryActiveMinutes | FairlyActiveMinutes | L |
|---|---|---|---|---|---|---|
| **845** | 8378563200 | 2016-05-09 | 8382 | 71 | 13 | |
| **112** | 1844505072 | 2016-05-02 | 0 | 0 | 0 | |
| **336** | 3977333714 | 2016-05-02 | 16520 | 24 | 143 | |
| **631** | 6290855005 | 2016-04-18 | 6885 | 0 | 0 | |
| **831** | 8378563200 | 2016-04-25 | 12405 | 117 | 16 | |
| **...** | ... | ... | ... | ... | ... | |
| **118** | 1844505072 | 2016-05-08 | 0 | 0 | 0 | |
| **830** | 8378563200 | 2016-04-24 | 3703 | 0 | 0 | |
| **328** | 3977333714 | 2016-04-24 | 14112 | 30 | 95 | |
| **111** | 1844505072 | 2016-05-01 | 2573 | 0 | 7 | |
| **469** | 4445114986 | 2016-05-08 | 7303 | 0 | 8 | |

940 rows × 11 columns

Next steps: Generate code with `daily_activity` View recommended plots

```
#last check of dataset we use this data set just only exploratory analyze but maybe our prep dataset use ML process.
#here we see if there's null data
daily_activity.isnull().sum()
```

```
Id                      0
ActivityDate            0
TotalSteps              0
VeryActiveMinutes       0
FairlyActiveMinutes     0
LightlyActiveMinutes    0
SedentaryMinutes        0
Calories                0
TotalMinutes            0
TotalHours              0
DayOfWeek               0
dtype: int64
```

```
#check about missing values
daily_activity.isna().sum()
```

```
Id                      0
ActivityDate            0
TotalSteps              0
VeryActiveMinutes       0
FairlyActiveMinutes     0
LightlyActiveMinutes    0
SedentaryMinutes        0
Calories                0
TotalMinutes            0
TotalHours              0
DayOfWeek               0
dtype: int64
```

```
daily_activity.duplicated().sum()
```

```
0
```

```
daily_activity.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 940 entries, 845 to 469
Data columns (total 11 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Id                    940 non-null    int64
 1   ActivityDate          940 non-null    datetime64[ns]
 2   TotalSteps            940 non-null    int64
 3   VeryActiveMinutes     940 non-null    int64
 4   FairlyActiveMinutes   940 non-null    int64
 5   LightlyActiveMinutes  940 non-null    int64
 6   SedentaryMinutes      940 non-null    int64
 7   Calories              940 non-null    int64
```

```
 8   TotalMinutes          940 non-null    int64
 9   TotalHours            940 non-null    float64
 10  DayOfWeek             940 non-null    category
dtypes: category(1), datetime64[ns](1), float64(1), int64(8)
memory usage: 82.0 KB
```

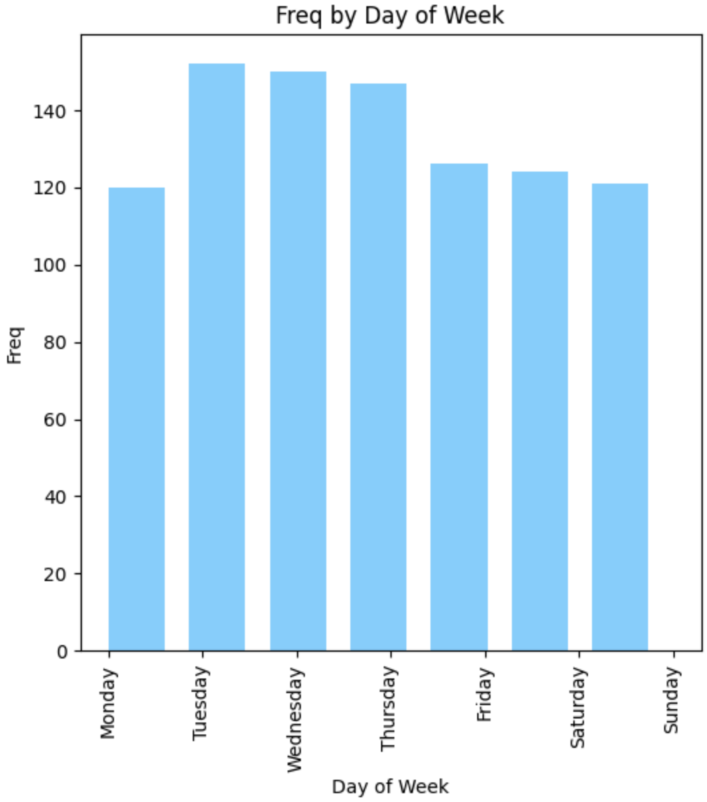
## Analysis and Visualization

```
daily_activity.describe()
```

| | Id | ActivityDate | TotalSteps | VeryActiveMinutes | FairlyActive |
|---|---|---|---|---|---|
| **count** | 9.400000e+02 | 940 | 940.000000 | 940.000000 | 940 |
| **mean** | 4.855407e+09 | 2016-04-26 06:53:37.021276672 | 7637.910638 | 21.164894 | 13 |
| **min** | 1.503960e+09 | 2016-04-12 00:00:00 | 0.000000 | 0.000000 | 0 |
| **25%** | 2.320127e+09 | 2016-04-19 00:00:00 | 3789.750000 | 0.000000 | 0 |
| **50%** | 4.445115e+09 | 2016-04-26 00:00:00 | 7405.500000 | 4.000000 | 6 |
| | | 2016-05-04 | | | |

```
plt.figure(figsize=(6, 6))
plt.hist(daily_activity.DayOfWeek, bins=7, color='lightskyblue', width=0.6)


plt.xlabel('Day of Week')
plt.ylabel('Freq')
plt.xticks(rotation = 90)
plt.title('Freq by Day of Week');
```




We notice that people are very active in tuesday, wednesday, and thursday, so we can send motivation message for people in the other days

```
daily_activity.info()
```

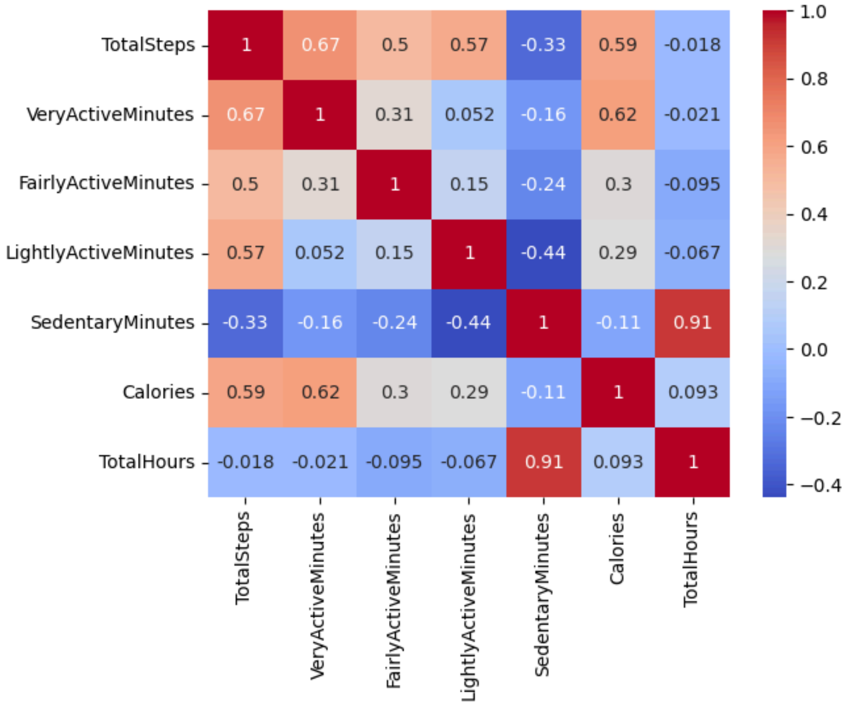
```
<class 'pandas.core.frame.DataFrame'>
Index: 940 entries, 845 to 469
Data columns (total 11 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
```

```
 0   Id                    940 non-null    int64
 1   ActivityDate          940 non-null    datetime64[ns]
 2   TotalSteps            940 non-null    int64
 3   VeryActiveMinutes     940 non-null    int64
 4   FairlyActiveMinutes   940 non-null    int64
 5   LightlyActiveMinutes  940 non-null    int64
 6   SedentaryMinutes      940 non-null    int64
 7   Calories              940 non-null    int64
 8   TotalMinutes          940 non-null    int64
 9   TotalHours            940 non-null    float64
 10  DayOfWeek             940 non-null    category
dtypes: category(1), datetime64[ns](1), float64(1), int64(8)
memory usage: 82.0 KB
```

```
# Define the Numerical Values to find the relationship between these columns and the calories column
numerical_columns_corr = daily_activity[["TotalSteps", "VeryActiveMinutes","FairlyActiveMinutes", "LightlyActiveMinutes", "SedentaryMin

# Here we display the corr between all columns
sns.heatmap(numerical_columns_corr,cmap="coolwarm",annot=True)
```

```
<Axes: >
```




from the heatmap we can notice that the TotalSteps and VeryActiveMintues Columns have the highest influence on the Calories column

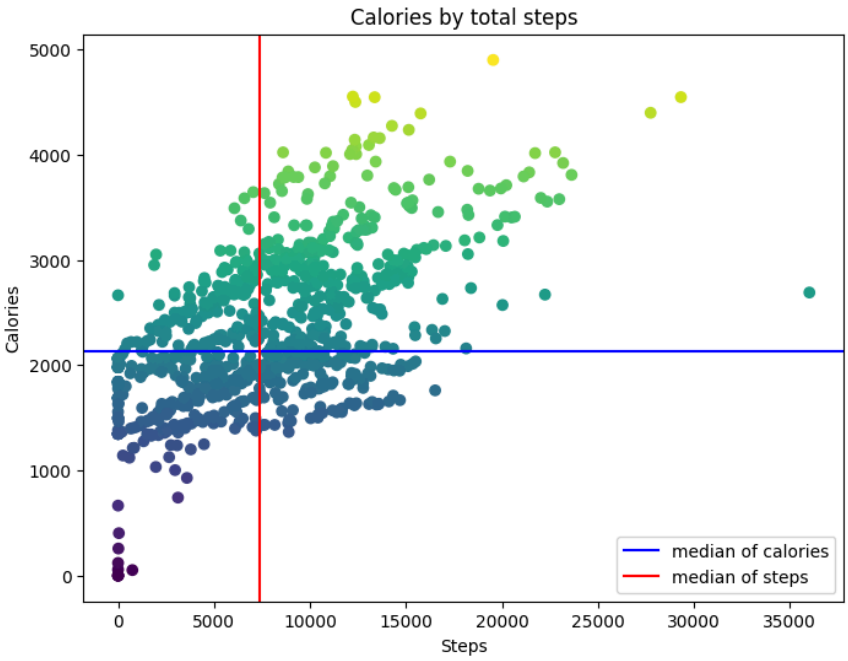
```
# Visualize the relationship between the TotalSteps column and the Calories column
plt.figure(figsize=(8, 6))
plt.scatter(daily_activity['TotalSteps'],daily_activity['Calories'],c = daily_activity['Calories'])

median_steps = 7405
median_calories = 2134

plt.axhline(median_calories, color = 'blue', label='median of calories')
plt.axvline(median_steps, color = 'red', label='median of steps')


plt.xlabel('Steps')
plt.ylabel('Calories')

plt.title('Calories by total steps');
plt.legend()
plt.show()
```



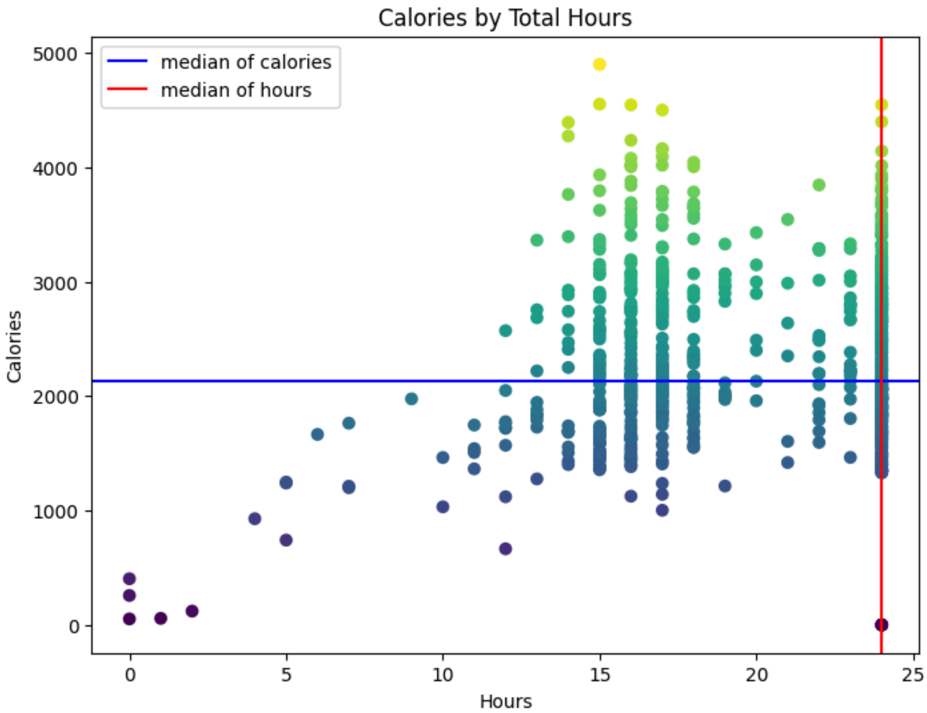

```
# Visualize the relationship between the TotalHours column and the Calories column
plt.figure(figsize=(8, 6))
plt.scatter(daily_activity['TotalHours'],daily_activity['Calories'],c = daily_activity['Calories'])

median_hours = 24
median_calories = 2134

plt.axhline(median_calories, color = 'blue', label='median of calories')
plt.axvline(median_hours, color = 'red', label='median of hours')


plt.xlabel('Hours')
plt.ylabel('Calories')

plt.title('Calories by Total Hours');
plt.legend()
plt.show()
```



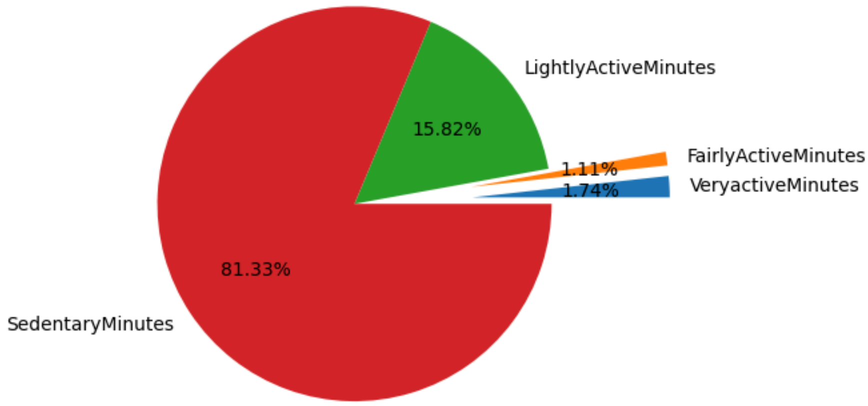

we notice that there is a weak relationship between them, and I think this happened because the few number of active minutes

```
# Visualize the percentage of each column of these columns [VeryActiveMinutes, FairlyActiveMinutes, LightlyActiveMinutes, SedentaryMinut
```

```
# visualize the percentage of each column of these columns (VeryActiveMinutes, FairlyActiveMinutes, LightlyActiveMinutes, SedentaryMinute
FairlyActiveMinutes = daily_activity['FairlyActiveMinutes'].sum()
VeryActiveMinutes = daily_activity['VeryActiveMinutes'].sum()
LightlyActiveMinutes = daily_activity['LightlyActiveMinutes'].sum()
SedentaryMinutes = daily_activity['SedentaryMinutes'].sum()

minutes = [VeryActiveMinutes, FairlyActiveMinutes, LightlyActiveMinutes, SedentaryMinutes]
labels = ['VeryactiveMinutes', 'FairlyActiveMinutes', 'LightlyActiveMinutes', 'SedentaryMinutes']

plt.pie(minutes, labels=labels, autopct='%1.2f%%', explode=[0.6, 0.6, 0, 0]);
```




We can say that most people use the company's products to calculate calories burned in normal daily activities such as walking to the market or to the bus stop, etc., and not sports activities such as running

```
# Sort the DataFrame by Country column in ascending order
sorted_calories_table = daily_activity.sort_values(by=['Calories'], ascending=False)

sorted_calories_table
```

| | Id | ActivityDate | TotalSteps | VeryActiveMinutes | FairlyActiveMinutes | LightlyActiveMinutes | SedentaryMinutes | Calories |
|---|---|---|---|---|---|---|---|---|
| **606** | 6117666160 | 2016-04-21 | 19542 | 11 | 19 | 294 | 579 | 4900 |
| **572** | 5577150313 | 2016-04-17 | 12231 | 200 | 37 | 159 | 525 | 4552 |
| **913** | 8877689391 | 2016-04-16 | 29326 | 94 | 29 | 429 | 888 | 4547 |
| **586** | 5577150313 | 2016-05-01 | 13368 | 194 | 72 | 178 | 499 | 4546 |
| **585** | 5577150313 | 2016-04-30 | 12363 | 207 | 45 | 163 | 621 | 4501 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **345** | 3977333714 | 2016-05-11 | 746 | 4 | 0 | 9 | 13 | 52 |
| **879** | 8583815059 | 2016-05-12 | 0 | 0 | 0 | 0 | 1440 | 0 |
| **653** | 6290855005 | 2016-05-10 | 0 | 0 | 0 | 0 | 1440 | 0 |
| **817** | 8253242879 | 2016-04-30 | 0 | 0 | 0 | 0 | 1440 | 0 |
| **30** | 1503960366 | 2016-05-12 | 0 | 0 | 0 | 0 | 1440 | 0 |

940 rows × 11 columns

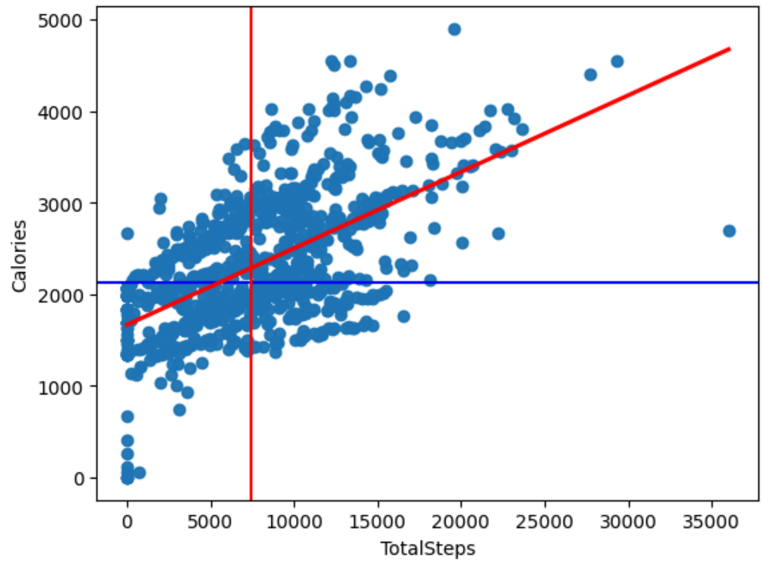
Next steps: Generate code with `sorted_calories_table` View recommended plots

```
from matplotlib import pyplot as plt
sorted_calories_table.plot(kind='scatter', x='TotalSteps', y='Calories', s=32, alpha=.8)
plt.gca().spines[['top', 'right',]].set_visible(True)

plt.axhline(median_calories, color = 'blue', label='median of calories')
plt.axvline(median_steps, color = 'red', label='median of steps')

sns.regplot(x=sorted_calories_table['TotalSteps'], y=sorted_calories_table['Calories'], ci=False, line_kws={'color':'red'})
```

```
<Axes: xlabel='TotalSteps', ylabel='Calories'>
```




# ACT

- Further analyses relating to customer segmentation still needed to be done to better understand customer archetypes, particularly on those who might be interested in improving their overall well-being. Bellabeat is well-poised to delivering a tech-based solution for this group of customers.
- Since our target population is those who have yet purchase a Bellabeat product, the marketing strategy have to include reasons to why customers should buy Bellabeat's wearables and use Bellabeat's app. Recall that our business task is to uncover insights into how consumers are using smart devices, not why consumers should purchase our smart devices. The former is what we were tasked to analyse but note that this differed from the ultimate goal of convincing prospective customers to purchase our products.
- Another minor point to point out is that the Bellabeat team is astute in how they positioned their products. Bellabeats unique selling point (USP) is how their products are focused on women, and they made a very convincing pitch to how Bellabeat is the smart device of choice for women, with women-focused products like reproductive health tracking and pregnancy tracking integrated into their suite of products.

---

## Let's back second dataset

```
#Before de analyzes I checked each table is part of dataset for maximum information
weightLogInfo = pd.read_csv('/content/drive/MyDrive/ColabNotebooks/Case_Study_2/weightLogInfo_merged.csv')
weightLogInfo
```

| | Id | Date | WeightKg | WeightPounds | Fat | BMI | IsManualReport | LogId |
|---|---|---|---|---|---|---|---|---|
| **0** | 1503960366 | 5/2/2016 11:59:59 PM | 52.599998 | 115.963147 | 22.0 | 22.650000 | True | 1462233599000 |
| **1** | 1503960366 | 5/3/2016 11:59:59 PM | 52.599998 | 115.963147 | NaN | 22.650000 | True | 1462319999000 |
| **2** | 1927972279 | 4/13/2016 1:08:52 AM | 133.500000 | 294.317120 | NaN | 47.540001 | False | 1460509732000 |
| **3** | 2873212765 | 4/21/2016 11:59:59 PM | 56.700001 | 125.002104 | NaN | 21.450001 | True | 1461283199000 |
| **4** | 2873212765 | 5/12/2016 11:59:59 PM | 57.299999 | 126.324875 | NaN | 21.690001 | True | 1463097599000 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **62** | 8877689391 | 5/6/2016 6:43:35 AM | 85.000000 | 187.392923 | NaN | 25.440001 | False | 1462517015000 |
| **63** | 8877689391 | 5/8/2016 7:35:53 AM | 85.400002 | 188.274775 | NaN | 25.559999 | False | 1462692953000 |
| **64** | 8877689391 | 5/9/2016 6:39:44 AM | 85.500000 | 188.495234 | NaN | 25.610001 | False | 1462775984000 |
| **65** | 8877689391 | 5/11/2016 6:51:47 AM | 85.400002 | 188.274775 | NaN | 25.559999 | False | 1462949507000 |
| **66** | 8877689391 | 5/12/2016 6:42:53 AM | 84.000000 | 185.188300 | NaN | 25.139999 | False | 1463035373000 |

67 rows × 8 columns

Next steps: Generate code with weightLogInfo | View recommended plots

```
weightLogInfo =weightLogInfo.sort_values(by= ['Id','WeightKg'], ascending=False)
weightLogInfo
```

| | Id | Date | WeightKg | WeightPounds | Fat | BMI | IsManualReport | LogId |
|---|---|---|---|---|---|---|---|---|
| 43 | 8877689391 | 4/12/2016 6:47:11 AM | 85.800003 | 189.156628 | NaN | 25.680000 | False | 1460443631000 |
| 47 | 8877689391 | 4/18/2016 6:51:14 AM | 85.800003 | 189.156628 | NaN | 25.680000 | False | 1460962274000 |
| 46 | 8877689391 | 4/16/2016 1:39:25 PM | 85.500000 | 188.495234 | NaN | 25.590000 | False | 1460813965000 |
| 51 | 8877689391 | 4/23/2016 7:22:28 AM | 85.500000 | 188.495234 | NaN | 25.590000 | False | 1461396148000 |
| 52 | 8877689391 | 4/24/2016 7:38:05 AM | 85.500000 | 188.495234 | NaN | 25.590000 | False | 1461483485000 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 4 | 2873212765 | 5/12/2016 11:59:59 PM | 57.299999 | 126.324875 | NaN | 21.690001 | True | 1463097599000 |
| 3 | 2873212765 | 4/21/2016 11:59:59 PM | 56.700001 | 125.002104 | NaN | 21.450001 | True | 1461283199000 |
| 2 | 1927972279 | 4/13/2016 1:08:52 AM | 133.500000 | 294.317120 | NaN | 47.540001 | False | 1460509732000 |
| 0 | 1503960366 | 5/2/2016 11:59:59 PM | 52.599998 | 115.963147 | 22.0 | 22.650000 | True | 1462233599000 |
| 1 | 1503960366 | 5/3/2016 11:59:59 PM | 52.599998 | 115.963147 | NaN | 22.650000 | True | 1462319999000 |

67 rows × 8 columns

Next steps: Generate code with weightLogInfo View recommended plots

```
# Which user follow own weight Bellabeat apps
weightLogInfo.Id.unique()
```

```
array([8877689391, 6962181067, 5577150313, 4558609924, 4319703577,
       2873212765, 1927972279, 1503960366])
```

```
columns2 = ['Id', 'Date', 'WeightKg']
weightLogInfo = weightLogInfo[columns2]
weightLogInfo
```

| | Id | Date | WeightKg |
|---|---|---|---|
| 43 | 8877689391 | 4/12/2016 6:47:11 AM | 85.800003 |
| 47 | 8877689391 | 4/18/2016 6:51:14 AM | 85.800003 |
| 46 | 8877689391 | 4/16/2016 1:39:25 PM | 85.500000 |
| 51 | 8877689391 | 4/23/2016 7:22:28 AM | 85.500000 |
| 52 | 8877689391 | 4/24/2016 7:38:05 AM | 85.500000 |
| ... | ... | ... | ... |
| 4 | 2873212765 | 5/12/2016 11:59:59 PM | 57.299999 |
| 3 | 2873212765 | 4/21/2016 11:59:59 PM | 56.700001 |
| 2 | 1927972279 | 4/13/2016 1:08:52 AM | 133.500000 |
| 0 | 1503960366 | 5/2/2016 11:59:59 PM | 52.599998 |
| 1 | 1503960366 | 5/3/2016 11:59:59 PM | 52.599998 |

67 rows × 3 columns

Next steps: Generate code with weightLogInfo View recommended plots

```
weightLogInfo.groupby(by=['Id'], dropna= False,).describe()
```

| | WeightKg | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | count | mean | std | min | 25% | 50% | 75% | max |
| Id | | | | | | | | |
| 1503960366 | 2.0 | 52.599998 | 0.000000 | 52.599998 | 52.599998 | 52.599998 | 52.599998 | 52.599998 |
| 1927972279 | 1.0 | 133.500000 | NaN | 133.500000 | 133.500000 | 133.500000 | 133.500000 | 133.500000 |
| 2873212765 | 2.0 | 57.000000 | 0.424263 | 56.700001 | 56.850000 | 57.000000 | 57.150000 | 57.299999 |
| 4319703577 | 2.0 | 72.350002 | 0.070710 | 72.300003 | 72.325003 | 72.350002 | 72.375002 | 72.400002 |
| 4558609924 | 5.0 | 69.639999 | 0.497998 | 69.099998 | 69.199997 | 69.699997 | 69.900002 | 70.300003 |
| 5577150313 | 1.0 | 90.699997 | NaN | 90.699997 | 90.699997 | 90.699997 | 90.699997 | 90.699997 |
| 6962181067 | 30.0 | 61.553334 | 0.388395 | 61.000000 | 61.225000 | 61.500000 | 61.700001 | 62.500000 |
| 8877689391 | 24.0 | 85.145834 | 0.454905 | 84.000000 | 84.900002 | 85.300003 | 85.500000 | 85.800003 |

From our users 6962181067 and 8877689391 use frequently our apps for health monitoring

```
# Infos of Id's 6962181067 user
user_6962181067 =weightLogInfo.loc[weightLogInfo['Id'] == 6962181067]
user_6962181067
```

| | Id | Date | WeightKg |
|---|---|---|---|
| 13 | 6962181067 | 4/12/2016 11:59:59 PM | 62.500000 |
| 39 | 6962181067 | 5/9/2016 11:59:59 PM | 62.400002 |
| 14 | 6962181067 | 4/13/2016 11:59:59 PM | 62.099998 |
| 40 | 6962181067 | 5/10/2016 11:59:59 PM | 62.099998 |
| 17 | 6962181067 | 4/16/2016 11:59:59 PM | 62.000000 |
| 41 | 6962181067 | 5/11/2016 11:59:59 PM | 61.900002 |
| 42 | 6962181067 | 5/12/2016 11:59:59 PM | 61.900002 |
| 15 | 6962181067 | 4/14/2016 11:59:59 PM | 61.700001 |
| 21 | 6962181067 | 4/20/2016 11:59:59 PM | 61.700001 |
| 26 | 6962181067 | 4/25/2016 11:59:59 PM | 61.700001 |
| 31 | 6962181067 | 5/1/2016 11:59:59 PM | 61.700001 |
| 16 | 6962181067 | 4/15/2016 11:59:59 PM | 61.500000 |
| 24 | 6962181067 | 4/23/2016 11:59:59 PM | 61.500000 |
| 25 | 6962181067 | 4/24/2016 11:59:59 PM | 61.500000 |
| 32 | 6962181067 | 5/2/2016 11:59:59 PM | 61.500000 |
| 36 | 6962181067 | 5/6/2016 11:59:59 PM | 61.500000 |
| 18 | 6962181067 | 4/17/2016 11:59:59 PM | 61.400002 |
| 20 | 6962181067 | 4/19/2016 11:59:59 PM | 61.400002 |
| 22 | 6962181067 | 4/21/2016 11:59:59 PM | 61.400002 |
| 23 | 6962181067 | 4/22/2016 11:59:59 PM | 61.400002 |
| 29 | 6962181067 | 4/29/2016 11:59:59 PM | 61.400002 |
| 35 | 6962181067 | 5/5/2016 11:59:59 PM | 61.299999 |
| 19 | 6962181067 | 4/18/2016 11:59:59 PM | 61.200001 |
| 27 | 6962181067 | 4/27/2016 11:59:59 PM | 61.200001 |
| 28 | 6962181067 | 4/28/2016 11:59:59 PM | 61.200001 |
| 37 | 6962181067 | 5/7/2016 11:59:59 PM | 61.200001 |
| 38 | 6962181067 | 5/8/2016 11:59:59 PM | 61.200001 |
| 34 | 6962181067 | 5/4/2016 11:59:59 PM | 61.099998 |
| 30 | 6962181067 | 4/30/2016 11:59:59 PM | 61.000000 |
| 33 | 6962181067 | 5/3/2016 11:59:59 PM | 61.000000 |

```
# Infos of Id's 8877689391 user
user_8877689391 =weightLogInfo.loc[weightLogInfo['Id'] == 8877689391]
user_8877689391
```

| | Id | Date | WeightKg |
|---|---|---|---|
| 43 | 8877689391 | 4/12/2016 6:47:11 AM | 85.800003 |
| 47 | 8877689391 | 4/18/2016 6:51:14 AM | 85.800003 |
| 46 | 8877689391 | 4/16/2016 1:39:25 PM | 85.500000 |
| 51 | 8877689391 | 4/23/2016 7:22:28 AM | 85.500000 |
| 52 | 8877689391 | 4/24/2016 7:38:05 AM | 85.500000 |
| 58 | 8877689391 | 4/30/2016 7:49:03 AM | 85.500000 |
| 64 | 8877689391 | 5/9/2016 6:39:44 AM | 85.500000 |
| 53 | 8877689391 | 4/25/2016 6:40:16 AM | 85.400002 |
| 55 | 8877689391 | 4/27/2016 6:51:05 AM | 85.400002 |
| 63 | 8877689391 | 5/8/2016 7:35:53 AM | 85.400002 |
| 65 | 8877689391 | 5/11/2016 6:51:47 AM | 85.400002 |
| 48 | 8877689391 | 4/19/2016 6:39:31 AM | 85.300003 |
| 59 | 8877689391 | 5/1/2016 8:47:49 AM | 85.300003 |
| 54 | 8877689391 | 4/26/2016 6:50:27 AM | 85.099998 |
| 56 | 8877689391 | 4/28/2016 6:50:03 AM | 85.099998 |
| 62 | 8877689391 | 5/6/2016 6:43:35 AM | 85.000000 |
| 44 | 8877689391 | 4/13/2016 6:55:00 AM | 84.900002 |
| 49 | 8877689391 | 4/20/2016 6:44:54 AM | 84.900002 |
| 57 | 8877689391 | 4/29/2016 6:49:55 AM | 84.900002 |
| 60 | 8877689391 | 5/3/2016 6:49:41 AM | 84.900002 |
| 45 | 8877689391 | 4/14/2016 6:48:43 AM | 84.500000 |
| 50 | 8877689391 | 4/21/2016 6:50:27 AM | 84.500000 |
| 61 | 8877689391 | 5/4/2016 6:48:22 AM | 84.400002 |
| 66 | 8877689391 | 5/12/2016 6:42:53 AM | 84.000000 |

```
#Merging two most user datas

special_two_user =pd.concat([user_8877689391, user_6962181067], axis=0)
special_two_user.head(10)
```

| | Id | Date | WeightKg |
|---|---|---|---|
| 43 | 8877689391 | 4/12/2016 6:47:11 AM | 85.800003 |
| 47 | 8877689391 | 4/18/2016 6:51:14 AM | 85.800003 |
| 46 | 8877689391 | 4/16/2016 1:39:25 PM | 85.500000 |
| 51 | 8877689391 | 4/23/2016 7:22:28 AM | 85.500000 |
| 52 | 8877689391 | 4/24/2016 7:38:05 AM | 85.500000 |
| 58 | 8877689391 | 4/30/2016 7:49:03 AM | 85.500000 |
| 64 | 8877689391 | 5/9/2016 6:39:44 AM | 85.500000 |
| 53 | 8877689391 | 4/25/2016 6:40:16 AM | 85.400002 |
| 55 | 8877689391 | 4/27/2016 6:51:05 AM | 85.400002 |
| 63 | 8877689391 | 5/8/2016 7:35:53 AM | 85.400002 |

```
# Sorted that Id and Date and group by Id thanks to that we save merged datas and users info together
special_two_user_sorted = (special_two_user.sort_values(by=['Id','Date'], ascending=False)).groupby('Id')
special_two_user_sorted.head(10)
```

| | Id | Date | WeightKg |
|---|---|---|---|
| **64** | 8877689391 | 5/9/2016 6:39:44 AM | 85.500000 |
| **63** | 8877689391 | 5/8/2016 7:35:53 AM | 85.400002 |
| **62** | 8877689391 | 5/6/2016 6:43:35 AM | 85.000000 |
| **61** | 8877689391 | 5/4/2016 6:48:22 AM | 84.400002 |
| **60** | 8877689391 | 5/3/2016 6:49:41 AM | 84.900002 |
| **66** | 8877689391 | 5/12/2016 6:42:53 AM | 84.000000 |
| **65** | 8877689391 | 5/11/2016 6:51:47 AM | 85.400002 |
| **59** | 8877689391 | 5/1/2016 8:47:49 AM | 85.300003 |
| **58** | 8877689391 | 4/30/2016 7:49:03 AM | 85.500000 |
| **57** | 8877689391 | 4/29/2016 6:49:55 AM | 84.900002 |
| **39** | 6962181067 | 5/9/2016 11:59:59 PM | 62.400002 |
| **38** | 6962181067 | 5/8/2016 11:59:59 PM | 61.200001 |