Hemalatha_Subbiah_Project_Final

Hemalatha Subbiah

2023-03-04

R Markdown

Introduction

An activity tracker is a type of electronic device that helps monitor some type of human activity, such as walking or running, sleep quality or heart rate.Better's research shows that almost three-quarters of people who wear fitness trackers do so to monitor their progress, while 62% of wearers use them to increase their motivation to exercise. Another 46% want to understand their body better by tracking things like their heart rate, steps taken and calories burned.Activity trackers are devices that translate movement into different forms of data. Most trackers will provide estimates of steps, distance, and active minutes.

Problem Statement

This data science project aims to help data scientists develop an intelligent model for how can your own personal analysis data assist you in living a better life? To solve this project related to data science, the popular Kaggle dataset containing activity tracker transaction made in September 2016 by individuals.This Kaggle data set contains personal fitness tracker from thirty activity tracker users. The dataset contains 18 .csv files, which is used are about activity, calories, intensity, steps, and sleep time. Activity tracker collect continuous physiological measurement and are generating gigabytes of data every single minute. Fitbit reports that they have over 150 billion hours of heart rate recorded, and over 6 billion recorded nights of human sleep. While this data is extremely useful for gathering information at the population-level, how can your own personal analysis assist you in living a better life? Do activity trackers really help to better your health?All the information exists at your fingertips (or on your wrist), and we can make it actionable

Research questions

1. Do people have Awareness to relate the data to personal health ?
2. What are some trends in smart device usage?
3. How could these trends apply to activity trackers customers?
4. How could these trends help influence good health Business task?
5. Identify potential opportunities for growth and recommendations for the Bellabeat marketing strategy improvement based on trends in smart device usage.
6. Is it focused own women from all countries?
7. IS tracking physical activity, mental state, menstrual cycle helps them to improve health better?
8. Is it possible to collect a large amount of data about personal activity relatively inexpensively?
9. Do people have Awareness to relate the data to personal health ?

Approach

Business understanding Generate Your Hypotheses Study the data Clean the data Engineer the features Model Fitting Making Prediction

How your approach addresses (fully or partially) the problem

Business understanding : In business understand phase we basically Understands the business process,Define and Frame the business problem,define the business objective and agree on success criteria.In my project how the activity trackers really help to better your health and the business task is to identify themes/trends in how people currently use their smart devices and relate to their own health.

Data understanding : Understand data touch points in the context of business process and gather knowledge on where data originates from, how it gets processed, what decisions are being made, where it is getting stored and how it flows to downstream.Deep dive into business meaning of the data being leveraged as well as knowledge present in existing system in form of rules. For this project this dataset would have been more reliable, original, current, and cited; albeit data privacy would have to be carefully guarded.However, since this is a hypothetical scenario and this is the only dataset available, I'll make do.

Data preparation and Cleaning :Good data hygiene is so important for business. For starters, it's good practice to keep on top of your data, ensuring that it's accurate and up-to-date.As part of data cleaning I want to a) Get rid of unwanted observations b) Fix structural errors - Removed inconsistent capitalization, which often occur during manual data entry c) Remove unwanted outliers- Removed few outliners in the data d) Fix contradictory data errors e) Type conversion and syntax errors f) Validate your dataset for null values and condition them. All the above steps will be completed as part of data sets I picked for this project.

Modeling :Build predictive model variables and do feature engineering and fit an closest model to the problem solution.

Validation : Validating the model by training the model. Deployment : The concept of deployment in data science refers to the application of a model for prediction using a new data. Building a model is generally not the end of the project. Even if the purpose of the model is to increase knowledge of the data, the knowledge gained will need to be organized and presented in a way that the customer can use it. Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data science process.

Data (Minimum of 3 Datasets - but no requirement on number of fields or rows)

The dataset used for this analysis is FitBit Fitness Tracker Data hosted on Kaggle or Zenodo. It comprises .csv files of various fitness metrics measured from different users at different times, stored in a wide format. The fitness tracker data was provided by 30 respondents to a paid distributed survey on Amazon Mechanical Turk in 2016.

Limitations of DataSet: Data was collected in 2016, hence data may not be relevant to modern trends. Small sample size of only 30 participants. Data does not include demographics about the sample such as sex, age, or geographical location. This may not be a good representation of the population of women globally who would use a similar product.Survey style of data collection may be subject to response bias. Integrity and accuracy of data is not clear.

Inital observations of these CSVs within Mircosoft Excel shows that these files contain acitvites, calory records, physical acitivity records, step record, sleep monitoring, heart rate, weight and BMI calculations.

Using simple unique formula against unique ID of users bring out the fact that these files contain the above mentioned data for anywhere between 8 to 33 users. Another point to be noted here is the fact that some of these numbers are manual input of users, such as weight in the weightLogInfo_merged.csv file.

```r
## Set the working directory to the root of your DSC 520 directory
## Load the `data/r4ds/week-6-housing.csv` to
setwd("C:/MastersCourse/RAssignemtents/data")

dailyActivity_data <- read.csv("dailyActivity_merged.csv", header = TRUE)
head(dailyActivity_data)
```

```
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
## 1 1503960366    4/12/2016      13162          8.50            8.50
## 2 1503960366    4/13/2016      10735          6.97            6.97
## 3 1503960366    4/14/2016      10460          6.74            6.74
## 4 1503960366    4/15/2016       9762          6.28            6.28
## 5 1503960366    4/16/2016      12669          8.16            8.16
## 6 1503960366    4/17/2016       9705          6.48            6.48
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               1.88                     0.55
## 2                        0               1.57                     0.69
## 3                        0               2.44                     0.40
## 4                        0               2.14                     1.26
## 5                        0               2.71                     0.41
## 6                        0               3.19                     0.78
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                6.06                       0                25
## 2                4.71                       0                21
## 3                3.91                       0                30
## 4                2.83                       0                29
## 5                5.04                       0                36
## 6                2.51                       0                38
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories
## 1                  13                  328              728     1985
## 2                  19                  217              776     1797
## 3                  11                  181             1218     1776
## 4                  34                  209              726     1745
## 5                  10                  221              773     1863
## 6                  20                  164              539     1728
```

```r
dailyCalories_data <- read.csv("dailyCalories_merged.csv", header = TRUE)
head(dailyActivity_data)
```

```
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
## 1 1503960366    4/12/2016      13162          8.50            8.50
## 2 1503960366    4/13/2016      10735          6.97            6.97
## 3 1503960366    4/14/2016      10460          6.74            6.74
## 4 1503960366    4/15/2016       9762          6.28            6.28
## 5 1503960366    4/16/2016      12669          8.16            8.16
## 6 1503960366    4/17/2016       9705          6.48            6.48
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               1.88                     0.55
## 2                        0               1.57                     0.69
```

```
## 3                     0              2.44                     0.40
## 4                     0              2.14                     1.26
## 5                     0              2.71                     0.41
## 6                     0              3.19                     0.78
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                6.06                       0                25
## 2                4.71                       0                21
## 3                3.91                       0                30
## 4                2.83                       0                29
## 5                5.04                       0                36
## 6                2.51                       0                38
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories
## 1                  13                  328              728     1985
## 2                  19                  217              776     1797
## 3                  11                  181             1218     1776
## 4                  34                  209              726     1745
## 5                  10                  221              773     1863
## 6                  20                  164              539     1728
```

```
dailySteps_data <- read.csv("dailySteps_merged.csv", header = TRUE)
head(dailySteps_data)
```

```
##           Id ActivityDay StepTotal
## 1 1503960366   4/12/2016     13162
## 2 1503960366   4/13/2016     10735
## 3 1503960366   4/14/2016     10460
## 4 1503960366   4/15/2016      9762
## 5 1503960366   4/16/2016     12669
## 6 1503960366   4/17/2016      9705
```

```
sleepDay_data <- read.csv("sleepDay_merged.csv", header = TRUE)
head(sleepDay_data)
```

```
##           Id              SleepDay TotalSleepRecords TotalMinutesAsleep
## 1 1503960366 4/12/2016 12:00:00 AM                 1                327
## 2 1503960366 4/13/2016 12:00:00 AM                 2                384
## 3 1503960366 4/15/2016 12:00:00 AM                 1                412
## 4 1503960366 4/16/2016 12:00:00 AM                 2                340
## 5 1503960366 4/17/2016 12:00:00 AM                 1                700
## 6 1503960366 4/19/2016 12:00:00 AM                 1                304
##   TotalTimeInBed
## 1            346
## 2            407
## 3            442
## 4            367
## 5            712
## 6            320
```

```
weightLogInfo_data <- read.csv("weightLogInfo_merged.csv", header = TRUE)
head(weightLogInfo_data)
```

```
##           Id                  Date WeightKg WeightPounds Fat   BMI
```

```
## 1 1503960366  5/2/2016 11:59:59 PM     52.6     115.9631  22 22.65
## 2 1503960366  5/3/2016 11:59:59 PM     52.6     115.9631  NA 22.65
## 3 1927972279  4/13/2016 1:08:52 AM    133.5     294.3171  NA 47.54
## 4 2873212765 4/21/2016 11:59:59 PM     56.7     125.0021  NA 21.45
## 5 2873212765 5/12/2016 11:59:59 PM     57.3     126.3249  NA 21.69
## 6 4319703577 4/17/2016 11:59:59 PM     72.4     159.6147  25 27.45
##   IsManualReport        LogId
## 1           True 1.462234e+12
## 2           True 1.462320e+12
## 3          False 1.460510e+12
## 4           True 1.461283e+12
## 5           True 1.463098e+12
## 6           True 1.460938e+12
```

Required Packages

library("tidyverse") library("car") library("ggplot") library("dplyr") library("ggplot2") library("tidyr") library("dply")

Plots and Table Needs

ggplot : histogram : Density Curves : Box plot : Line Plot : Scatter Diagram :

Questions for future steps :

1.A time series analysis of your heart rate to forecast your future heart rate and comparing it with normal healthy heart rate may lead to think of healthy lie style.

2. What changes to dietary and life style choices after watching the data ?

3.Predicting cholestrol depending on factors like calorie in take, weight, no. of steps walked every day, distance covered, heart rate bpm, types of type of physical excercise,Although one of these activities you have to track outside of Activity Trackers. What all more factors as you see fit?

4.What are the effects of using these devices and correlating them to Relationship satisfaction or quality of life?

5.What type of decisions will our data science feature drive?

6.What metric will we use to call this project a success and how will we measure it?

7.what do they currently use and what is the baseline (current) value of that metric?

8.What the outcome of this project success?

How to import and clean my data :

```r
## Set the working directory to the root of your DSC 520 directory
## Load the `data/r4ds/week-6-housing.csv` to
setwd("C:/MastersCourse/RAssignemtents/data")

dailyActivity_data <- read.csv("dailyActivity_merged.csv", header = TRUE)
head(dailyActivity_data)
```

```
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
## 1 1503960366    4/12/2016      13162          8.50            8.50
## 2 1503960366    4/13/2016      10735          6.97            6.97
## 3 1503960366    4/14/2016      10460          6.74            6.74
## 4 1503960366    4/15/2016       9762          6.28            6.28
## 5 1503960366    4/16/2016      12669          8.16            8.16
## 6 1503960366    4/17/2016       9705          6.48            6.48
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               1.88                     0.55
## 2                        0               1.57                     0.69
## 3                        0               2.44                     0.40
## 4                        0               2.14                     1.26
## 5                        0               2.71                     0.41
## 6                        0               3.19                     0.78
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                6.06                       0                25
## 2                4.71                       0                21
## 3                3.91                       0                30
## 4                2.83                       0                29
## 5                5.04                       0                36
## 6                2.51                       0                38
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories
## 1                  13                  328              728     1985
## 2                  19                  217              776     1797
## 3                  11                  181             1218     1776
## 4                  34                  209              726     1745
## 5                  10                  221              773     1863
## 6                  20                  164              539     1728
```

```r
dailyCalories_data <- read.csv("dailyCalories_merged.csv", header = TRUE)
head(dailyCalories_data)
```

```
##           Id ActivityDay Calories
## 1 1503960366   4/12/2016     1985
## 2 1503960366   4/13/2016     1797
## 3 1503960366   4/14/2016     1776
## 4 1503960366   4/15/2016     1745
## 5 1503960366   4/16/2016     1863
## 6 1503960366   4/17/2016     1728
```

```r
dailySteps_data <- read.csv("dailySteps_merged.csv", header = TRUE)
head(dailySteps_data)
```

```
##           Id ActivityDay StepTotal
## 1 1503960366   4/12/2016     13162
```

```
## 2 1503960366   4/13/2016     10735
## 3 1503960366   4/14/2016     10460
## 4 1503960366   4/15/2016      9762
## 5 1503960366   4/16/2016     12669
## 6 1503960366   4/17/2016      9705
```

```
sleepDay_data <- read.csv("sleepDay_merged.csv", header = TRUE)
head(sleepDay_data)
```

```
##           Id              SleepDay TotalSleepRecords TotalMinutesAsleep
## 1 1503960366 4/12/2016 12:00:00 AM                 1                327
## 2 1503960366 4/13/2016 12:00:00 AM                 2                384
## 3 1503960366 4/15/2016 12:00:00 AM                 1                412
## 4 1503960366 4/16/2016 12:00:00 AM                 2                340
## 5 1503960366 4/17/2016 12:00:00 AM                 1                700
## 6 1503960366 4/19/2016 12:00:00 AM                 1                304
##   TotalTimeInBed
## 1            346
## 2            407
## 3            442
## 4            367
## 5            712
## 6            320
```

```
weightLogInfo_data <- read.csv("weightLogInfo_merged.csv", header = TRUE)
head(weightLogInfo_data)
```

```
##           Id                  Date WeightKg WeightPounds Fat   BMI
## 1 1503960366  5/2/2016 11:59:59 PM     52.6     115.9631  22 22.65
## 2 1503960366  5/3/2016 11:59:59 PM     52.6     115.9631  NA 22.65
## 3 1927972279  4/13/2016 1:08:52 AM    133.5     294.3171  NA 47.54
## 4 2873212765 4/21/2016 11:59:59 PM     56.7     125.0021  NA 21.45
## 5 2873212765 5/12/2016 11:59:59 PM     57.3     126.3249  NA 21.69
## 6 4319703577 4/17/2016 11:59:59 PM     72.4     159.6147  25 27.45
##   IsManualReport        LogId
## 1           True 1.462234e+12
## 2           True 1.462320e+12
## 3          False 1.460510e+12
## 4           True 1.461283e+12
## 5           True 1.463098e+12
## 6           True 1.460938e+12
```

1. Clean column names

2.Remove empty column or rows Suppose if you want to remove the column or row if contain completely empty, then you can use remove_empty function.

3. To Exclude duplicates and remove null:

```
dailyActivity_data <- unique(dailyActivity_data)                                      # Exclude duplica
dailyCalories_data <- unique(dailyCalories_data)
dailySteps_data <- unique(dailySteps_data)
sleepDay_data <- unique(sleepDay_data)
weightLogInfo_data <- unique(weightLogInfo_data)

sum(is.na(dailyActivity_data))
```

```
## [1] 0
```

```
sum(is.na(dailyCalories_data))
```

```
## [1] 0
```

```
sum(is.na(dailySteps_data))
```

```
## [1] 0
```

```
sum(is.na(sleepDay_data))
```

```
## [1] 0
```

```
sum(is.na(weightLogInfo_data))
```

```
## [1] 65
```

4. Convert data type

```
sapply(dailyActivity_data, class)
```

```
##                       Id             ActivityDate               TotalSteps
##                "numeric"              "character"                "integer"
##            TotalDistance          TrackerDistance LoggedActivitiesDistance
##                "numeric"                "numeric"                "numeric"
##       VeryActiveDistance ModeratelyActiveDistance      LightActiveDistance
##                "numeric"                "numeric"                "numeric"
##  SedentaryActiveDistance        VeryActiveMinutes      FairlyActiveMinutes
##                "numeric"                "integer"                "integer"
##     LightlyActiveMinutes         SedentaryMinutes                 Calories
##                "integer"                "integer"                "integer"
```

```
sapply(dailyCalories_data, class)
```

```
##          Id ActivityDay    Calories
##   "numeric" "character"   "integer"
```

```r
sapply(dailySteps_data, class)
```

```
##          Id ActivityDay   StepTotal
##   "numeric" "character"   "integer"
```

```r
sapply(sleepDay_data, class)
```

```
##                  Id           SleepDay  TotalSleepRecords TotalMinutesAsleep
##           "numeric"        "character"          "integer"          "integer"
##      TotalTimeInBed
##           "integer"
```

```r
sapply(weightLogInfo_data, class)
```

```
##              Id            Date        WeightKg    WeightPounds             Fat
##       "numeric"     "character"       "numeric"       "numeric"       "integer"
##             BMI IsManualReport           LogId
##       "numeric"     "character"       "numeric"
```

```r
dailyActivity_data <- type.convert(dailyActivity_data, as.is = TRUE)
dailyCalories_data <- type.convert(dailyCalories_data, as.is = TRUE)
dailySteps_data <- type.convert(dailySteps_data, as.is = TRUE)
sleepDay_data <- type.convert(sleepDay_data, as.is = TRUE)
weightLogInfo_data <- type.convert(weightLogInfo_data, as.is = TRUE)
```

5. Detect & Remove Outliers

```r
weightLogInfo_data$weight_kg[weightLogInfo_data$weight_kg%in% boxplot.stats(weightLogInfo_data$weight_kg
```

```
## NULL
```

```r
head(dailyActivity_data)
```

```
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
## 1 1503960366    4/12/2016      13162          8.50            8.50
## 2 1503960366    4/13/2016      10735          6.97            6.97
## 3 1503960366    4/14/2016      10460          6.74            6.74
## 4 1503960366    4/15/2016       9762          6.28            6.28
## 5 1503960366    4/16/2016      12669          8.16            8.16
## 6 1503960366    4/17/2016       9705          6.48            6.48
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               1.88                     0.55
## 2                        0               1.57                     0.69
## 3                        0               2.44                     0.40
## 4                        0               2.14                     1.26
## 5                        0               2.71                     0.41
## 6                        0               3.19                     0.78
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                6.06                       0                25
```

```
## 2                4.71                        0                21
## 3                3.91                        0                30
## 4                2.83                        0                29
## 5                5.04                        0                36
## 6                2.51                        0                38
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories
## 1                  13                  328              728     1985
## 2                  19                  217              776     1797
## 3                  11                  181             1218     1776
## 4                  34                  209              726     1745
## 5                  10                  221              773     1863
## 6                  20                  164              539     1728
```

What does the final data set look like?

It might be difficult to understand at first what the data means and what column names to use, but after couple of analysis was able to figure out the data.

#preparing daily activity dataset,

```
library(lubridate)
```

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

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

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

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

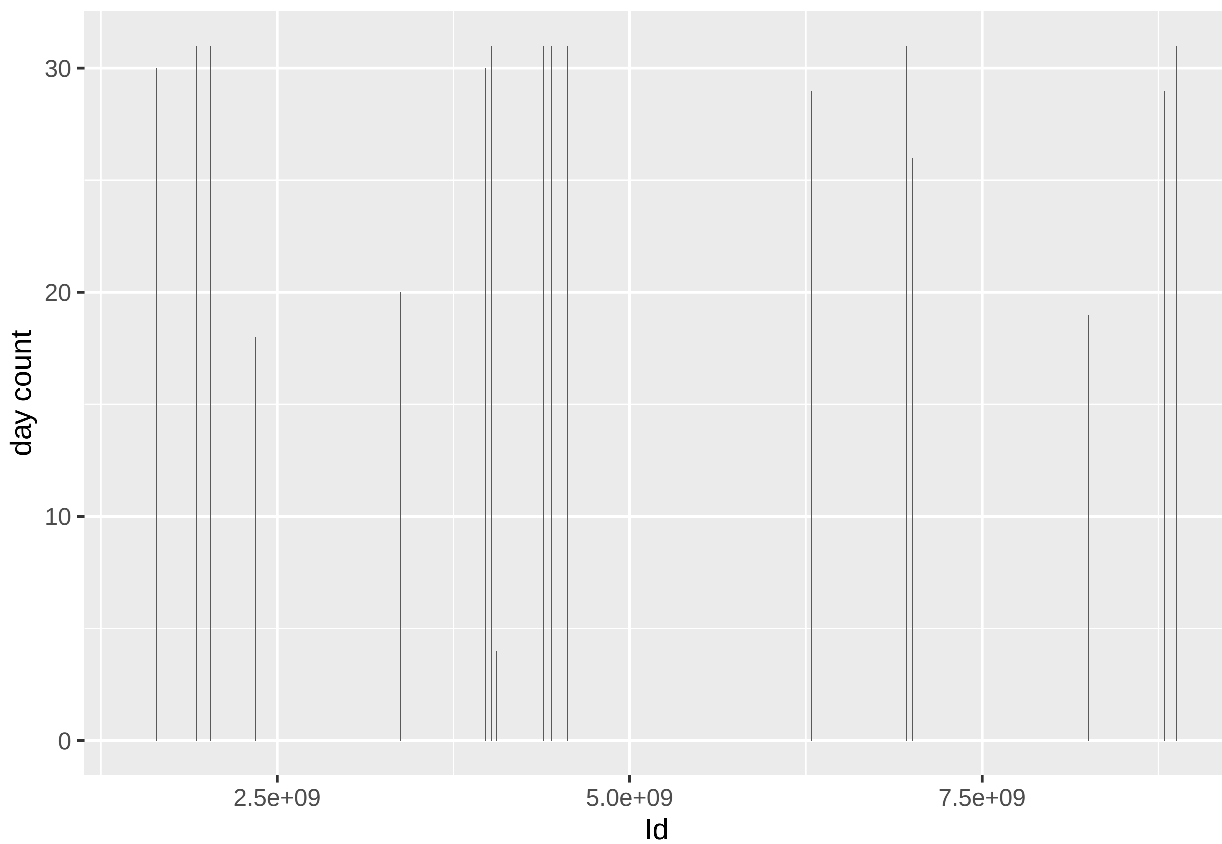
```
library(ggplot2)
dailyActivity_data <- dailyActivity_data %>% mutate(day = as.Date(ActivityDate)) %>% select(-c(2,5:9))
print(paste(c("Rows: ", "Columns: "), dim(dailyActivity_data)))
```

```
## [1] "Rows:  940"   "Columns:  10"
```

```
dailyactivity <- distinct(dailyActivity_data)
print(paste(c("Rows: ", "Columns: "), dim(dailyActivity_data)))
```

```
## [1] "Rows:  940"   "Columns:  10"
```

```
ggplot(data=dailyActivity_data,aes(x=Id)) +
  geom_bar() + ylab("day count")
```




```
library(lubridate)
library(dplyr)
library(ggplot2)
head(weightLogInfo_data)
```

```
##           Id                  Date WeightKg WeightPounds Fat   BMI
## 1 1503960366  5/2/2016 11:59:59 PM     52.6     115.9631  22 22.65
## 2 1503960366  5/3/2016 11:59:59 PM     52.6     115.9631  NA 22.65
## 3 1927972279  4/13/2016 1:08:52 AM    133.5     294.3171  NA 47.54
## 4 2873212765 4/21/2016 11:59:59 PM     56.7     125.0021  NA 21.45
## 5 2873212765 5/12/2016 11:59:59 PM     57.3     126.3249  NA 21.69
## 6 4319703577 4/17/2016 11:59:59 PM     72.4     159.6147  25 27.45
##   IsManualReport        LogId
## 1           True 1.462234e+12
## 2           True 1.462320e+12
## 3          False 1.460510e+12
```

```
## 4           True 1.461283e+12
## 5           True 1.463098e+12
## 6           True 1.460938e+12
```

```
weightLogInfo_data <- weightLogInfo_data %>% mutate (time = mdy_hms(as.Date(Date)))%>% mutate(day = dat
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `time = mdy_hms(as.Date(Date))`.
## Caused by warning:
## ! All formats failed to parse. No formats found.
```

```
print(paste(c("Rows: ", "Columns: "), dim(weightLogInfo_data)))
```

```
## [1] "Rows:  67"   "Columns:  6"
```

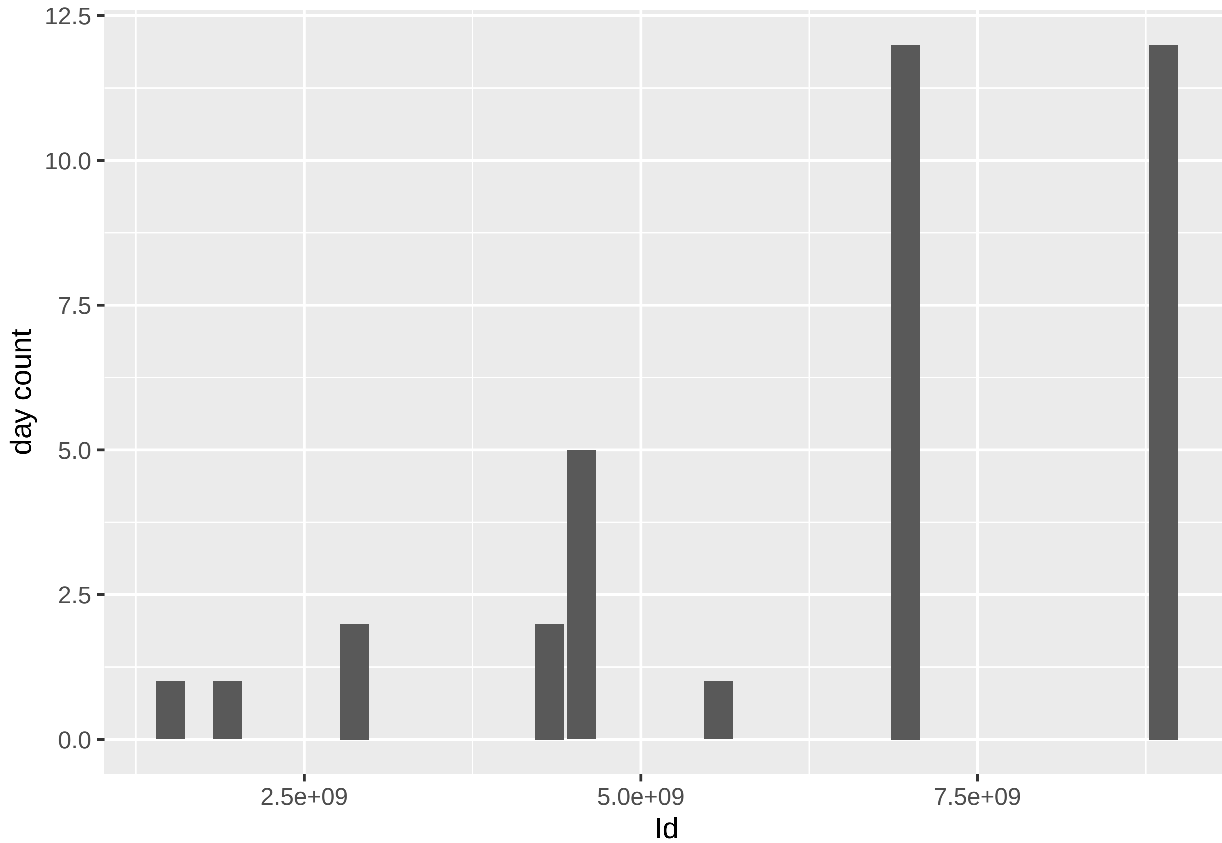
```
weightLogInfo_data <- distinct(weightLogInfo_data)
print(paste(c("Rows: ", "Columns: "), dim(weightLogInfo_data)))
```

```
## [1] "Rows:  36"   "Columns:  6"
```

```
head(weightLogInfo_data)
```

```
##           Id WeightKg WeightPounds   BMI IsManualReport  day
## 1 1503960366     52.6     115.9631 22.65           True <NA>
## 2 1927972279    133.5     294.3171 47.54          False <NA>
## 3 2873212765     56.7     125.0021 21.45           True <NA>
## 4 2873212765     57.3     126.3249 21.69           True <NA>
## 5 4319703577     72.4     159.6147 27.45           True <NA>
## 6 4319703577     72.3     159.3942 27.38           True <NA>
```

```
ggplot(data=weightLogInfo_data, aes(x=Id)) +
  geom_bar() + ylab("day count")
```



Merging data

Before beginning to visualize the data, I need to merge two data sets. I'm going to merge (inner join) activity and sleep on columns Id and date (that I previously created after converting data to date time format).

```
merged_data <- merge(sleepDay_data, dailyActivity_data,  by=c('Id'))
head(merged_data)
```

```
##           Id              SleepDay TotalSleepRecords TotalMinutesAsleep
## 1 1503960366 4/12/2016 12:00:00 AM                 1                327
## 2 1503960366 4/12/2016 12:00:00 AM                 1                327
## 3 1503960366 4/12/2016 12:00:00 AM                 1                327
## 4 1503960366 4/12/2016 12:00:00 AM                 1                327
## 5 1503960366 4/12/2016 12:00:00 AM                 1                327
## 6 1503960366 4/12/2016 12:00:00 AM                 1                327
##   TotalTimeInBed TotalSteps TotalDistance SedentaryActiveDistance
## 1            346      11992          7.71                       0
## 2            346      12159          8.03                       0
## 3            346      10602          6.81                       0
## 4            346      14673          9.25                       0
## 5            346      13162          8.50                       0
## 6            346      10735          6.97                       0
##   VeryActiveMinutes FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes
## 1                37                  46                  175              833
```

```
## 2                24                  6                  289              754
## 3                33                 35                  246              730
## 4                52                 34                  217              712
## 5                25                 13                  328              728
## 6                21                 19                  217              776
##   Calories        day
## 1     1821 0005-07-20
## 2     1896 0005-06-20
## 3     1820 0005-01-20
## 4     1947       <NA>
## 5     1985 0004-12-20
## 6     1797       <NA>
```

```
merged_data <- merge(dailyActivity_data,  weightLogInfo_data, by=c('Id'))
head(merged_data)
```

```
##           Id TotalSteps TotalDistance SedentaryActiveDistance VeryActiveMinutes
## 1 1503960366      12669          8.16                       0                36
## 2 1503960366      13019          8.59                       0                42
## 3 1503960366       9762          6.28                       0                29
## 4 1503960366      10060          6.58                       0                44
## 5 1503960366       9705          6.48                       0                38
## 6 1503960366      15506          9.88                       0                50
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories      day.x
## 1                  10                  221              773     1863       <NA>
## 2                  16                  233             1149     1921       <NA>
## 3                  34                  209              726     1745       <NA>
## 4                   8                  203              574     1740 0005-08-20
## 5                  20                  164              539     1728       <NA>
## 6                  31                  264              775     2035       <NA>
##   WeightKg WeightPounds   BMI IsManualReport day.y
## 1     52.6     115.9631 22.65           True  <NA>
## 2     52.6     115.9631 22.65           True  <NA>
## 3     52.6     115.9631 22.65           True  <NA>
## 4     52.6     115.9631 22.65           True  <NA>
## 5     52.6     115.9631 22.65           True  <NA>
## 6     52.6     115.9631 22.65           True  <NA>
```

```
#joining the essential data frames earlier read above
head(dailyCalories_data)
```

```
##           Id ActivityDay Calories
## 1 1503960366   4/12/2016     1985
## 2 1503960366   4/13/2016     1797
## 3 1503960366   4/14/2016     1776
## 4 1503960366   4/15/2016     1745
## 5 1503960366   4/16/2016     1863
## 6 1503960366   4/17/2016     1728
```

```
all_tables <- dailyActivity_data %>% full_join(sleepDay_data, by = c("Id")) %>% full_join(dailyCalories_
```

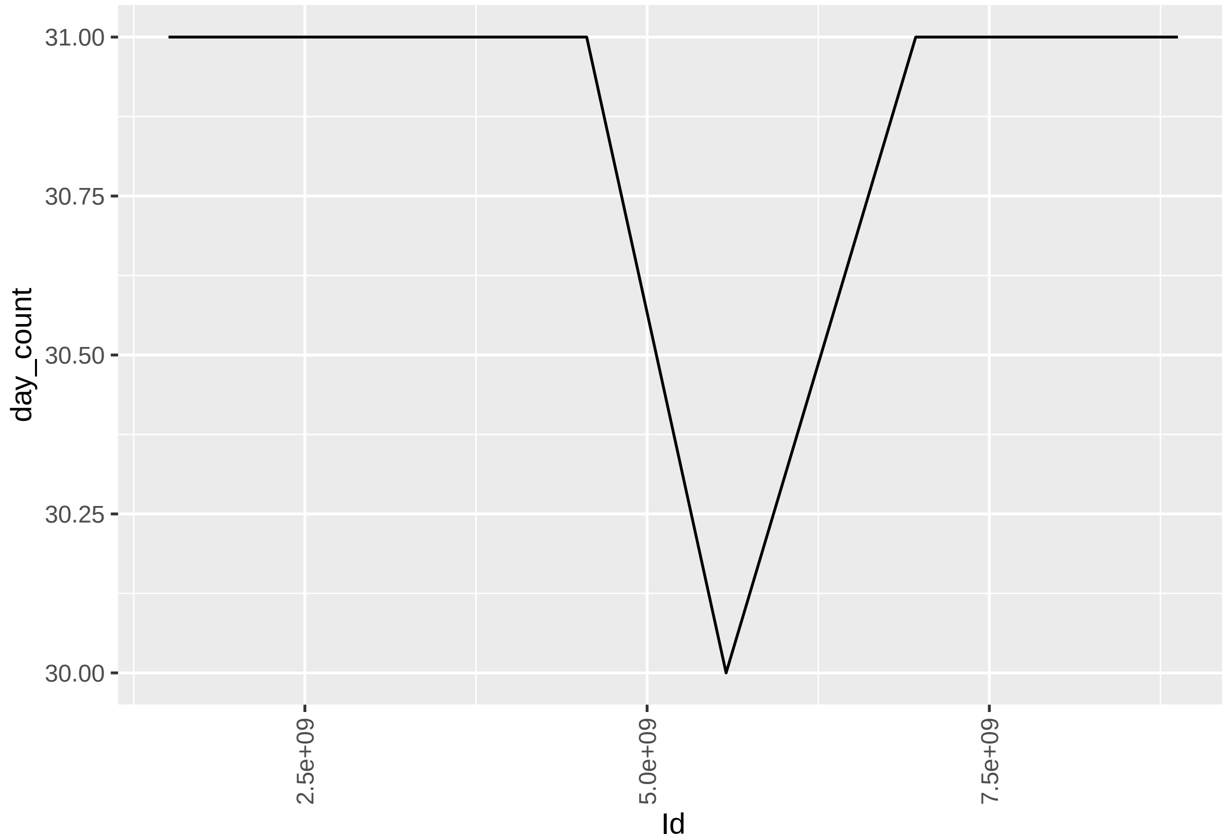
```
## Warning in full_join(., sleepDay_data, by = c("Id")): Each row in `x` is expected to match at most 1
```

```
## i Row 1 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.

## Warning in full_join(., dailyCalories_data, by = c("Id")): Each row in `x` is expected to match at mo
## i Row 1 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.

## Warning in full_join(., weightLogInfo_data, by = c("Id")): Each row in `x` is expected to match at mo
## i Row 69965 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```

```
head(all_tables)
```

```
##           Id TotalSteps TotalDistance SedentaryActiveDistance VeryActiveMinutes
## 1 1503960366      13162           8.5                       0                25
## 2 1503960366      13162           8.5                       0                25
## 3 1503960366      13162           8.5                       0                25
## 4 1503960366      13162           8.5                       0                25
## 5 1503960366      13162           8.5                       0                25
## 6 1503960366      13162           8.5                       0                25
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories.x
## 1                  13                  328              728       1985
## 2                  13                  328              728       1985
## 3                  13                  328              728       1985
## 4                  13                  328              728       1985
## 5                  13                  328              728       1985
## 6                  13                  328              728       1985
##        day.x              SleepDay TotalSleepRecords TotalMinutesAsleep
## 1 0004-12-20 4/12/2016 12:00:00 AM                 1                327
## 2 0004-12-20 4/12/2016 12:00:00 AM                 1                327
## 3 0004-12-20 4/12/2016 12:00:00 AM                 1                327
## 4 0004-12-20 4/12/2016 12:00:00 AM                 1                327
## 5 0004-12-20 4/12/2016 12:00:00 AM                 1                327
## 6 0004-12-20 4/12/2016 12:00:00 AM                 1                327
##   TotalTimeInBed ActivityDay Calories.y WeightKg WeightPounds   BMI
## 1            346   4/12/2016       1985     52.6     115.9631 22.65
## 2            346   4/13/2016       1797     52.6     115.9631 22.65
## 3            346   4/14/2016       1776     52.6     115.9631 22.65
## 4            346   4/15/2016       1745     52.6     115.9631 22.65
## 5            346   4/16/2016       1863     52.6     115.9631 22.65
## 6            346   4/17/2016       1728     52.6     115.9631 22.65
##   IsManualReport day.y
## 1           True  <NA>
## 2           True  <NA>
## 3           True  <NA>
## 4           True  <NA>
## 5           True  <NA>
## 6           True  <NA>
```

```
#day count for users who tracked their calories, checking frequency of daily use
calories<-filter(all_tables,BMI!=0)%>% group_by(Id)%>% summarise(day_count= n_distinct(ActivityDay), .g
ggplot(data=calories, mapping=aes(x=Id, y=day_count)) + geom_line()+ theme(axis.text.x = element_text(a
```



From the analysis I did the people are generally using fitness tracker to track activities and calories burned.And used it for sleep ans rest as well/

What information is not self-evident?

1.Key demographics data such as gender, age, were not identified. This is a crucial missing how far women use activity trackers.

2.User's exercise habits differ between summer and winter as the data is just for 31 days limited.

3.Health and lifestyle data is varied across different facets of society ,the data set is collected from small sample size.

What are different ways you could look at this data?

When I started analyzing the data, I want to set clear goals and expectations for what I wanted to learn and what insights you were expecting to find. I see that outliers in the data may skew the results. Significant outliers can easily skew averages in the data, so I may need to track the median rather than the mean. The median uses the middle value of the numerical data set, so it's less skewed by outliers. Alternatively, I may need to discount these outliers from your analysis altogether.

How do you plan to slice and dice the data?

Slicing means filtering rows from the data set and dicing means select set of columns from the data set.

With daily_activity data set,we will assume that days with < 200 TotalSteps taken, are days where users have not used their watches. We will filter out these inactive day and assign the following designations:

Low Use - 1 to 14 days o Moderate Use - 15 to 21 days High Use - 22 to 31 days

```
#data transformation to create df for 'Usage Types'
dailyActivity_data_group <- dailyActivity_data %>%
  filter(TotalSteps >200 ) %>%
  group_by(Id) %>%
  summarize(ActivityDate=sum(n())) %>%
  mutate(Usage = case_when(
    ActivityDate >= 1 & ActivityDate <= 14 ~ "Low Use",
    ActivityDate >= 15 & ActivityDate <= 21 ~ "Moderate Use",
    ActivityDate >= 22 & ActivityDate <= 31 ~ "High Use")) %>%
  mutate(Usage = factor(Usage, level = c('Low Use','Moderate Use','High Use'))) %>%
  rename(daysused = ActivityDate) %>%
  group_by(Usage)
head(dailyActivity_data_group)
```

```
## # A tibble: 6 x 3
## # Groups:   Usage [2]
##           Id daysused Usage
##        <dbl>    <int> <fct>
## 1 1503960366       30 High Use
## 2 1624580081       31 High Use
## 3 1644430081       30 High Use
## 4 1844505072       17 Moderate Use
## 5 1927972279       15 Moderate Use
## 6 2022484408       31 High Use
```

```
daily_use <- dailyActivity_data %>%
  left_join(dailyActivity_data_group, by = 'Id') %>%
  group_by(Usage) %>%
  summarise(participants = n_distinct(Id)) %>%
  mutate(perc = participants/sum(participants)) %>%
  arrange(perc) %>%
  mutate(perc = scales::percent(perc))
head(daily_use)
```

```
## # A tibble: 3 x 3
##   Usage        participants perc
##   <fct>               <int> <chr>
## 1 Low Use                 2 6%
## 2 Moderate Use            7 21%
## 3 High Use               24 73%
```

Above analysis ascertain how often the participants use their watches.We will filter out these how the activity trackers are used by the participant and categorized in to 3 parts.

How could you summarize your data to answer key questions?

```
# activity
dailyActivity_data %>%
  select(TotalSteps,
         TotalDistance,
         SedentaryMinutes, Calories) %>%
  summary()
```

```
##    TotalSteps    TotalDistance    SedentaryMinutes    Calories
##  Min.   :    0   Min.   : 0.000   Min.   :   0.0   Min.   :   0
##  1st Qu.: 3790   1st Qu.: 2.620   1st Qu.: 729.8   1st Qu.:1828
##  Median : 7406   Median : 5.245   Median :1057.5   Median :2134
##  Mean   : 7638   Mean   : 5.490   Mean   : 991.2   Mean   :2304
##  3rd Qu.:10727   3rd Qu.: 7.713   3rd Qu.:1229.5   3rd Qu.:2793
##  Max.   :36019   Max.   :28.030   Max.   :1440.0   Max.   :4900
```

```
# explore num of active minutes per category
dailyActivity_data %>%
  select(VeryActiveMinutes, FairlyActiveMinutes, LightlyActiveMinutes) %>%
  summary()
```

```
##  VeryActiveMinutes FairlyActiveMinutes LightlyActiveMinutes
##  Min.   :  0.00    Min.   :  0.00      Min.   :  0.0
##  1st Qu.:  0.00    1st Qu.:  0.00      1st Qu.:127.0
##  Median :  4.00    Median :  6.00      Median :199.0
##  Mean   : 21.16    Mean   : 13.56      Mean   :192.8
##  3rd Qu.: 32.00    3rd Qu.: 19.00      3rd Qu.:264.0
##  Max.   :210.00    Max.   :143.00      Max.   :518.0
```

```
# calories
dailyCalories_data %>%
  select(Calories) %>%
  summary()
```

```
##     Calories
##  Min.   :   0
##  1st Qu.:1828
##  Median :2134
##  Mean   :2304
##  3rd Qu.:2793
##  Max.   :4900
```

```
# sleep
sleepDay_data %>%
  select(TotalSleepRecords, TotalMinutesAsleep, TotalTimeInBed) %>%
  summary()
```

```
##  TotalSleepRecords TotalMinutesAsleep TotalTimeInBed
##  Min.   :1.00      Min.   : 58.0      Min.   : 61.0
```

```
##  1st Qu.:1.00      1st Qu.:361.0      1st Qu.:403.8  
##  Median :1.00      Median :432.5      Median :463.0  
##  Mean   :1.12      Mean   :419.2      Mean   :458.5  
##  3rd Qu.:1.00      3rd Qu.:490.0      3rd Qu.:526.0  
##  Max.   :3.00      Max.   :796.0      Max.   :961.0
```

```
# weight
weightLogInfo_data %>%
  select(WeightKg, BMI) %>%
  summary()
```

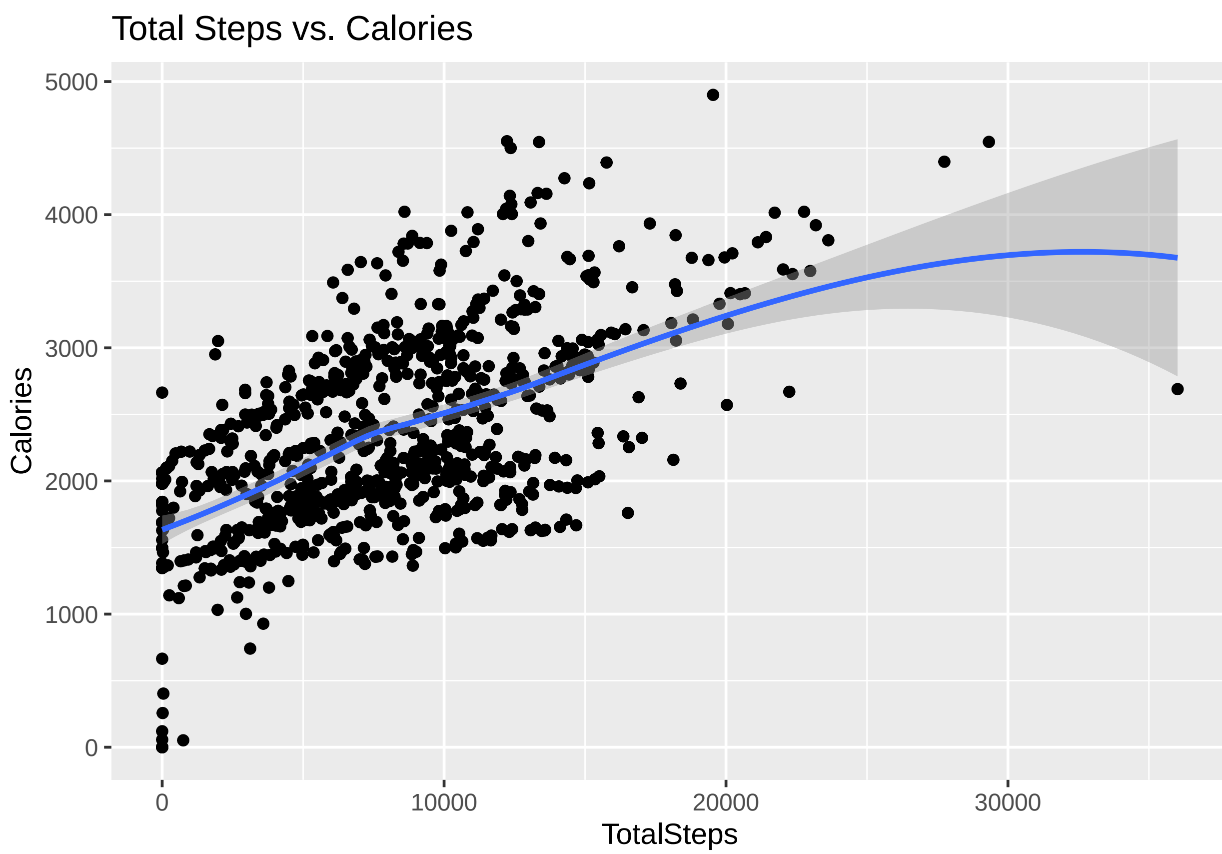
```
##     WeightKg           BMI       
##  Min.   : 52.60   Min.   :21.45  
##  1st Qu.: 61.65   1st Qu.:24.07  
##  Median : 69.80   Median :25.30  
##  Mean   : 73.44   Mean   :25.73  
##  3rd Qu.: 84.92   3rd Qu.:26.01  
##  Max.   :133.50   Max.   :47.54
```

Total Average step per day is 7638.They found that taking 8,000 steps per day was associated with a 51% lower risk for all-cause mortality (or death from all causes). Taking 12,000 steps per day was associated with a 65% lower risk compared with taking 4,000 steps. The majority of the participants are lightly active.

What types of plots and tables will help you to illustrate the findings to your questions?

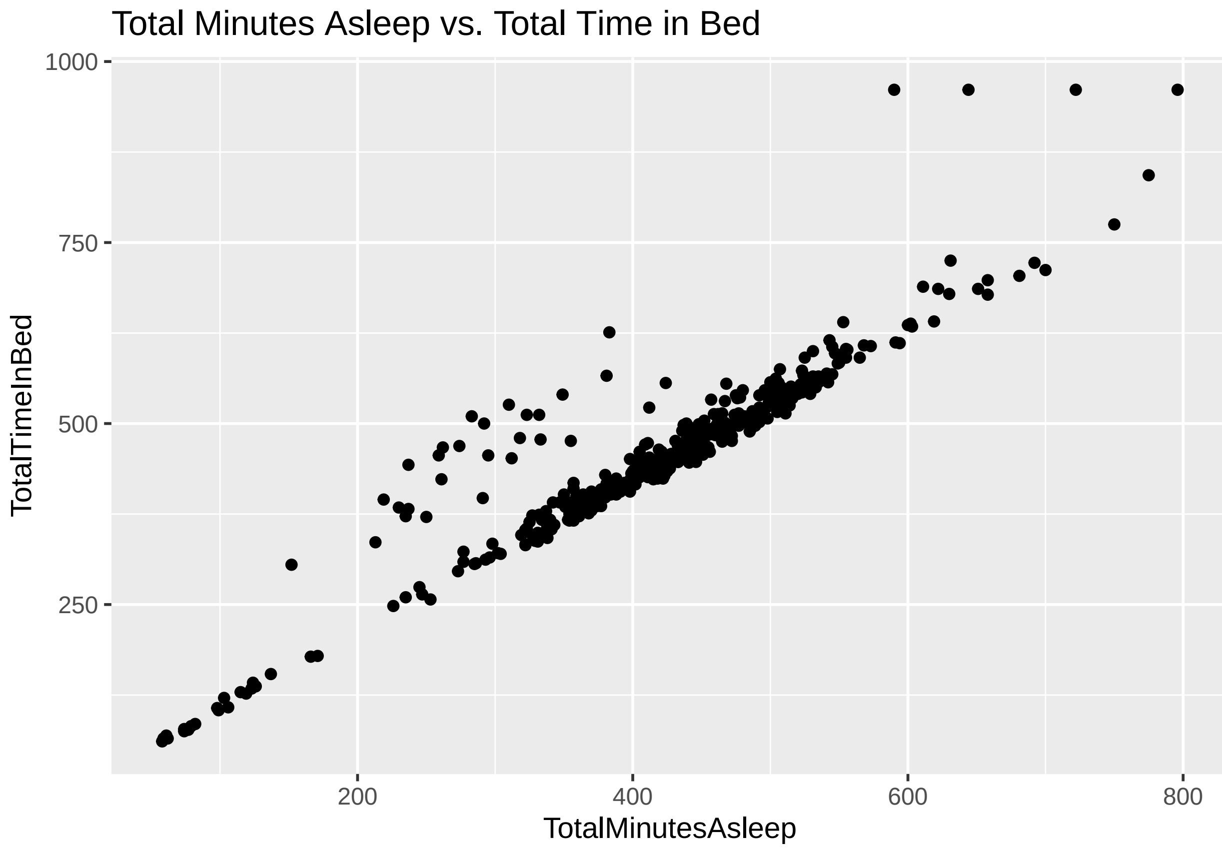
```
ggplot(data=dailyActivity_data, aes(x=TotalSteps, y=Calories)) +
  geom_point() + geom_smooth() + labs(title="Total Steps vs. Calories")
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



I see positive correlation here between Total Steps and Calories, which is obvious - the more active we are, the more calories we burn.

```
ggplot(data=sleepDay_data, aes(x=TotalMinutesAsleep, y=TotalTimeInBed)) +
  geom_point()+ labs(title="Total Minutes Asleep vs. Total Time in Bed")
```



The relationship between Total Minutes Asleep and Total Time in Bed looks linear. So if the Activity tracker users want to improve their sleep, we should consider using notification to go to sleep.

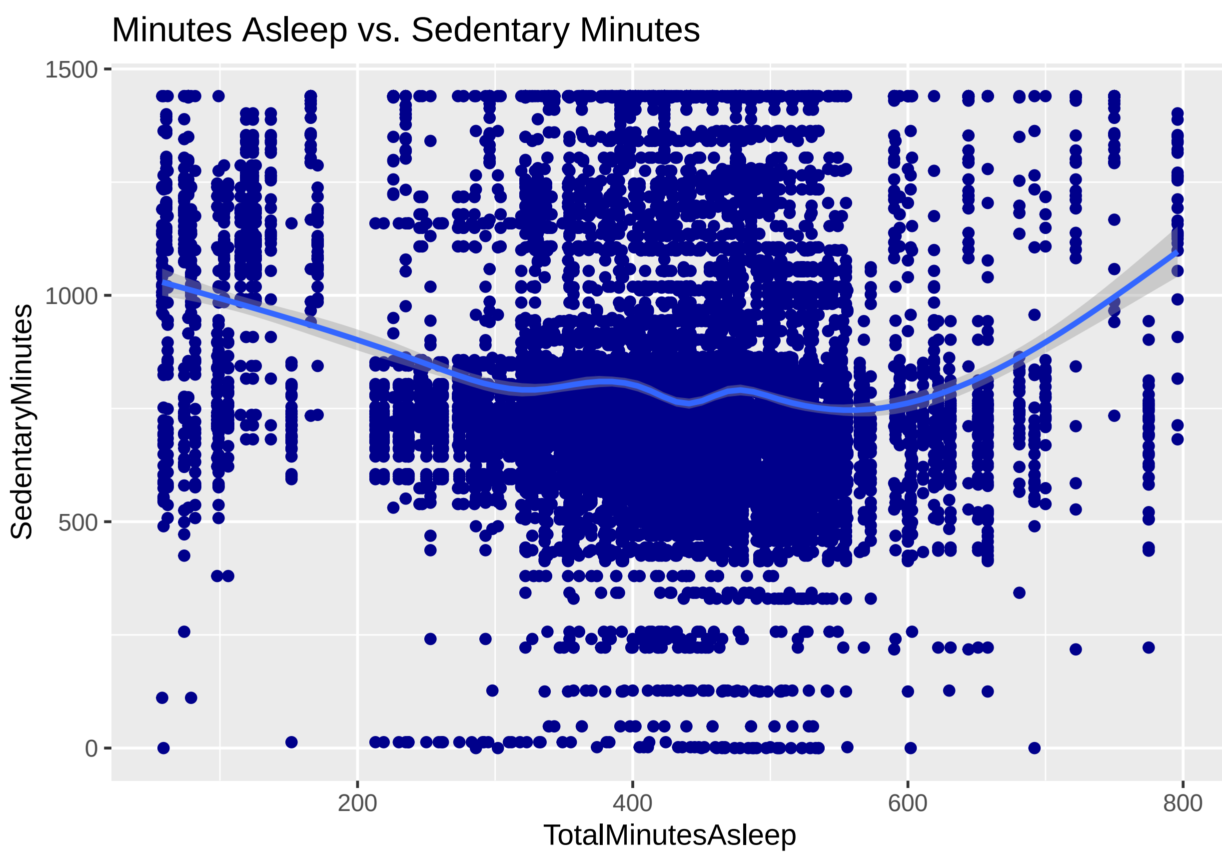
```
merged_data <- merge(sleepDay_data, dailyActivity_data,  by=c('Id'))
head(merged_data)
```

```
##           Id              SleepDay TotalSleepRecords TotalMinutesAsleep
## 1 1503960366 4/12/2016 12:00:00 AM                 1                327
## 2 1503960366 4/12/2016 12:00:00 AM                 1                327
## 3 1503960366 4/12/2016 12:00:00 AM                 1                327
## 4 1503960366 4/12/2016 12:00:00 AM                 1                327
## 5 1503960366 4/12/2016 12:00:00 AM                 1                327
## 6 1503960366 4/12/2016 12:00:00 AM                 1                327
##   TotalTimeInBed TotalSteps TotalDistance SedentaryActiveDistance
## 1            346      11992          7.71                       0
## 2            346      12159          8.03                       0
## 3            346      10602          6.81                       0
## 4            346      14673          9.25                       0
## 5            346      13162          8.50                       0
## 6            346      10735          6.97                       0
##   VeryActiveMinutes FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes
## 1                37                  46                  175              833
## 2                24                   6                  289              754
## 3                33                  35                  246              730
## 4                52                  34                  217              712
## 5                25                  13                  328              728
```

```
## 6                21                    19                    217              776
##   Calories        day
## 1     1821 0005-07-20
## 2     1896 0005-06-20
## 3     1820 0005-01-20
## 4     1947       <NA>
## 5     1985 0004-12-20
## 6     1797       <NA>
```

```
ggplot(data=merged_data, aes(x=TotalMinutesAsleep, y=SedentaryMinutes)) +
geom_point(color='darkblue') + geom_smooth() +
  labs(title="Minutes Asleep vs. Sedentary Minutes")
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



Do you plan on incorporating any machine learning techniques to answer your research questions? Explain :

Machine learning uses two techniques: supervised learning, which trains a model on known input and output data to predict future outputs, and unsupervised learning, which uses hidden patterns or internal structures in the input data.

In my use case its using Supervised machine learning creates a model that makes predictions based on evidence in the presence of uncertainty. A supervised learning algorithm takes a known set of input data and known responses to the data (output) and trains a model to generate reasonable predictions for the response to the new data. Use supervised learning if you have known data for the output you are trying to estimate.

Planning to use classification and regression techniques to develop machine learning models for my use case.

Regression analysis is used to estimate the relationship between a set of variables. When conducting any type of regression analysis, you're looking to see if there's a correlation between a dependent variable (that's the variable or outcome you want to measure or predict) and any number of independent variables (factors which may have an impact on the dependent variable). The aim of regression analysis is to estimate how one or more variables might impact the dependent variable, in order to identify trends and patterns. This is especially useful for making predictions and forecasting future trends.

Questions for future steps :

1.What are the effects of using these devices and correlating them to Relationship satisfaction or quality of life?

2. What changes to dietary and life style choices after watching the data ?

3.Predicting cholestrol depending on factors like calorie in take, weight, no. of steps walked every day, distance covered, heart rate bpm, types of type of physical excercise,Although one of these activities you have to track outside of Activity Trackers. What all more factors as you see fit?

4.What type of decisions will our data science feature drive?

5.What metric will we use to call this project a success and how will we measure it?

6.what do they currently use and what is the baseline (current) value of that metric?

7.What the outcome of this project success?

Introduction.

An activity tracker is a type of electronic device that helps monitor some type of human activity, such as walking or running, sleep quality or heart rate.Better's research shows that almost three-quarters of people who wear fitness trackers do so to monitor their progress, while 62% of wearers use them to increase their motivation to exercise. Another 46% want to understand their body better by tracking things like their heart rate, steps taken and calories burned.Activity trackers are devices that translate movement into different forms of data. Most trackers will provide estimates of steps, distance, and active minutes.

The problem statement you addressed.

This data science project aims to help data scientists develop an intelligent model for how can your own personal analysis data assist you in living a better life? To solve this project related to data science, the popular Kaggle dataset containing activity tracker transaction made in September 2016 by individuals.This Kaggle data set contains personal fitness tracker from thirty activity tracker users. The dataset contains 18 .csv files, which is used are about activity, calories, intensity, steps, and sleep time. Activity tracker collect continuous physiological measurement and are generating gigabytes of data every single minute. Fitbit reports that they have over 150 billion hours of heart rate recorded, and over 6 billion recorded nights of human sleep. While this data is extremely useful for gathering information at the population-level, how can your own personal analysis assist you in living a better life? Do activity trackers really help to better your health?All the information exists at your fingertips (or on your wrist), and we can make it actionable.

How you addressed this problem statement

I followed the below workflow to address the problem statement and derive the solution.Understanding and framing the problem is the first step of the data science life cycle.Found above problem statement which interested me about the activity tracker. The next step is to collect the right set of data. High-quality, targeted data—and the mechanisms to collect them—are crucial to obtaining meaningful results. Since much of the roughly 2.5 quintillion bytes of data created every day come in unstructured formats, you'll likely need to extract the data and export it into a usable format, such as a CSV or JSON file. Used popular Kaggle data set containing activity tracker transaction made in September 2016 by individuals. Most of the data you collect during the collection phase will be unstructured, irrelevant, and unfiltered. Bad data produces bad results, so the accuracy and efficacy of your analysis will depend heavily on the quality of your data.Cleaning data eliminates duplicate and null values,corrupt data,inconsistent data types, invalid entries, missing data, and improper formatting.Followed methods available in R to clean the the null values,duplicates and unwanted datas. Did Exploratory Data Analysis on data helped me to look at data before making any assumptions.It helped me to identify obvious errors, as well as better understand patterns within the data, detect outliners or anomalous events, find interesting relations among the variables.on the EDA can help answer questions about standard deviations, categorical variables, and confidence intervals. Once EDA is completed and insights are drawn,used its features for sophisticated data analysis or modeling,including machine learning. Then I tried doing Model Analysis where I planned to use machine learning, statistical models,and algorithms to extract high-value insights and predictions.

Analysis.

```r
## Set the working directory to the root of your DSC 520 directory
## Load the `data/r4ds/week-6-housing.csv` to
setwd("C:/MastersCourse/RAssignemtents/data")

dailyActivity_data <- read.csv("dailyActivity_merged.csv", header = TRUE)
head(dailyActivity_data)
```

```
##           Id ActivityDate TotalSteps TotalDistance TrackerDistance
## 1 1503960366    4/12/2016      13162          8.50            8.50
## 2 1503960366    4/13/2016      10735          6.97            6.97
## 3 1503960366    4/14/2016      10460          6.74            6.74
## 4 1503960366    4/15/2016       9762          6.28            6.28
## 5 1503960366    4/16/2016      12669          8.16            8.16
## 6 1503960366    4/17/2016       9705          6.48            6.48
##   LoggedActivitiesDistance VeryActiveDistance ModeratelyActiveDistance
## 1                        0               1.88                     0.55
## 2                        0               1.57                     0.69
## 3                        0               2.44                     0.40
## 4                        0               2.14                     1.26
## 5                        0               2.71                     0.41
## 6                        0               3.19                     0.78
##   LightActiveDistance SedentaryActiveDistance VeryActiveMinutes
## 1                6.06                       0                25
## 2                4.71                       0                21
## 3                3.91                       0                30
## 4                2.83                       0                29
## 5                5.04                       0                36
## 6                2.51                       0                38
```

```
##   FairlyActiveMinutes LightlyActiveMinutes SedentaryMinutes Calories
## 1                  13                  328              728     1985
## 2                  19                  217              776     1797
## 3                  11                  181             1218     1776
## 4                  34                  209              726     1745
## 5                  10                  221              773     1863
## 6                  20                  164              539     1728
```

```r
dailyCalories_data <- read.csv("dailyCalories_merged.csv", header = TRUE)
head(dailyCalories_data)
```

```
##           Id ActivityDay Calories
## 1 1503960366   4/12/2016     1985
## 2 1503960366   4/13/2016     1797
## 3 1503960366   4/14/2016     1776
## 4 1503960366   4/15/2016     1745
## 5 1503960366   4/16/2016     1863
## 6 1503960366   4/17/2016     1728
```

```r
dailySteps_data <- read.csv("dailySteps_merged.csv", header = TRUE)
head(dailySteps_data)
```

```
##           Id ActivityDay StepTotal
## 1 1503960366   4/12/2016     13162
## 2 1503960366   4/13/2016     10735
## 3 1503960366   4/14/2016     10460
## 4 1503960366   4/15/2016      9762
## 5 1503960366   4/16/2016     12669
## 6 1503960366   4/17/2016      9705
```

```r
sleepDay_data <- read.csv("sleepDay_merged.csv", header = TRUE)
head(sleepDay_data)
```

```
##           Id              SleepDay TotalSleepRecords TotalMinutesAsleep
## 1 1503960366 4/12/2016 12:00:00 AM                 1                327
## 2 1503960366 4/13/2016 12:00:00 AM                 2                384
## 3 1503960366 4/15/2016 12:00:00 AM                 1                412
## 4 1503960366 4/16/2016 12:00:00 AM                 2                340
## 5 1503960366 4/17/2016 12:00:00 AM                 1                700
## 6 1503960366 4/19/2016 12:00:00 AM                 1                304
##   TotalTimeInBed
## 1            346
## 2            407
## 3            442
## 4            367
## 5            712
## 6            320
```

```r
weightLogInfo_data <- read.csv("weightLogInfo_merged.csv", header = TRUE)
head(weightLogInfo_data)
```

```
##           Id                  Date WeightKg WeightPounds Fat   BMI
## 1 1503960366  5/2/2016 11:59:59 PM     52.6     115.9631  22 22.65
## 2 1503960366  5/3/2016 11:59:59 PM     52.6     115.9631  NA 22.65
## 3 1927972279  4/13/2016 1:08:52 AM    133.5     294.3171  NA 47.54
## 4 2873212765 4/21/2016 11:59:59 PM     56.7     125.0021  NA 21.45
## 5 2873212765 5/12/2016 11:59:59 PM     57.3     126.3249  NA 21.69
## 6 4319703577 4/17/2016 11:59:59 PM     72.4     159.6147  25 27.45
##   IsManualReport        LogId
## 1           True 1.462234e+12
## 2           True 1.462320e+12
## 3          False 1.460510e+12
## 4           True 1.461283e+12
## 5           True 1.463098e+12
## 6           True 1.460938e+12
```

I did analysis on usage distribution.Here we will ascertain how often the participants use their watches. With daily_activity, we will assume that days with < 200 TotalSteps taken, are days where users have not used their watches. We will filter out these inactive day and assign the following designations:

Low Use - 1 to 14 days o Moderate Use - 15 to 21 days High Use - 22 to 31 days

Average Steps By Hour, Day & Usage Types verage hourly steps increases as usage of devices increases across Usage Groups.

'High Use' group start their day an hour earlier (6:00AM) compared to other groups. Maintaining a higher average hourly step across all days of the week. Peaks in steps taken are consistently high between 5:00 - 8:00PM, suggesting habitual excercise as work ends.

'Moderate Use' group display peaks in their steps between 11:00AM - 12:00PM, and a rise between 6:00PM - 7:00PM.

'Low Use' group does not seem to display any symmetrical distribution of steps on any day of the week. This could be attributed to data gaps due to infrequent use.

Saturday is the most active day across all Usage Groups.

The 'High Use' group in general have a higher median calorie range compared to the upper quartile of moderate users. They also have a wider range the calories burnt each day of the week, displaying high variability of activities between users in this group. They are also much more consistent in carrying out physical activities throughout the week. No showing a bias on which days to be active.

The ' Moderate Use' group shows consistency in their daily average calorie burn. Saturdays are the most active day during week, displaying a preference to be active on this day

'High Use' participants have the most 'Lightly Active', 'Fairly Active' and 'Very Active' minutes across all groups. Subsequently, the spend the least time, 78.7% being sedentary. Unsurprisingly, thw 'Low Use' participants spend 93.2% of their day being sedentary.

Correlation study : we plot out the distribution of sleep for all participants based on the number of hours of sleep recommended by the National Sleep Foundation:

```r
library("dplyr")
library("tidyr")
sleepday2 <- sleepDay_data %>%
  select(TotalMinutesAsleep) %>%
  drop_na() %>%
  mutate(sleep_quality = ifelse(TotalMinutesAsleep <= 360, 'Below Recommended',
                         ifelse(TotalMinutesAsleep <= 420,  'Fairly Recommended',
                         ifelse(TotalMinutesAsleep <= 540, 'Recommended', 'Above Recommended')))) %>%
```

```r
  mutate(sleep_quality = factor(sleep_quality,level = c('Below Recommended', 'Fairly Recommended',
                                    'Recommended','Above Recommended')))
head(sleepday2)
```

```
##   TotalMinutesAsleep      sleep_quality
## 1                327  Below Recommended
## 2                384 Fairly Recommended
## 3                412 Fairly Recommended
## 4                340  Below Recommended
## 5                700  Above Recommended
## 6                304  Below Recommended
```

Below Recommended - < 6 hours of sleep Fairly Recommended - 6 and 7 hours of sleep Recommended - 7 and 9 hours of sleep Above Recommended - > 9 hours of sleep

```r
#joining the essential data frames earlier read above
head(dailyCalories_data)
```

```
##           Id ActivityDay Calories
## 1 1503960366   4/12/2016     1985
## 2 1503960366   4/13/2016     1797
## 3 1503960366   4/14/2016     1776
## 4 1503960366   4/15/2016     1745
## 5 1503960366   4/16/2016     1863
## 6 1503960366   4/17/2016     1728
```

```r
all_tables <- dailyActivity_data %>% full_join(sleepDay_data, by = c("Id")) %>% full_join(dailyCalories_
```

```
## Warning in full_join(., sleepDay_data, by = c("Id")): Each row in `x` is expected to match at most 1
## i Row 1 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.

## Warning in full_join(., dailyCalories_data, by = c("Id")): Each row in `x` is expected to match at mo
## i Row 1 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.

## Warning in full_join(., weightLogInfo_data, by = c("Id")): Each row in `x` is expected to match at mo
## i Row 1 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```

```r
str(all_tables)
```

```
## 'data.frame':    1339046 obs. of  28 variables:
##  $ Id                      : num  1.5e+09 1.5e+09 1.5e+09 1.5e+09 1.5e+09 ...
##  $ ActivityDate            : chr  "4/12/2016" "4/12/2016" "4/12/2016" "4/12/2016" ...
##  $ TotalSteps              : int  13162 13162 13162 13162 13162 13162 13162 13162 13162 13162 ...
##  $ TotalDistance           : num  8.5 8.5 8.5 8.5 8.5 8.5 8.5 8.5 8.5 8.5 ...
```

```
##  $ TrackerDistance         : num  8.5 8.5 8.5 8.5 8.5 8.5 8.5 8.5 8.5 8.5 ...
##  $ LoggedActivitiesDistance: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveDistance      : num  1.88 1.88 1.88 1.88 1.88 ...
##  $ ModeratelyActiveDistance: num  0.55 0.55 0.55 0.55 0.55 ...
##  $ LightActiveDistance     : num  6.06 6.06 6.06 6.06 6.06 ...
##  $ SedentaryActiveDistance : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ VeryActiveMinutes       : int  25 25 25 25 25 25 25 25 25 25 ...
##  $ FairlyActiveMinutes     : int  13 13 13 13 13 13 13 13 13 13 ...
##  $ LightlyActiveMinutes    : int  328 328 328 328 328 328 328 328 328 328 ...
##  $ SedentaryMinutes        : int  728 728 728 728 728 728 728 728 728 728 ...
##  $ Calories.x              : int  1985 1985 1985 1985 1985 1985 1985 1985 1985 1985 ...
##  $ SleepDay                : chr  "4/12/2016 12:00:00 AM" "4/12/2016 12:00:00 AM" "4/12/2016 12:00:0
##  $ TotalSleepRecords       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ TotalMinutesAsleep      : int  327 327 327 327 327 327 327 327 327 327 ...
##  $ TotalTimeInBed          : int  346 346 346 346 346 346 346 346 346 346 ...
##  $ ActivityDay             : chr  "4/12/2016" "4/12/2016" "4/13/2016" "4/13/2016" ...
##  $ Calories.y              : int  1985 1985 1797 1797 1776 1776 1745 1745 1863 1863 ...
##  $ Date                    : chr  "5/2/2016 11:59:59 PM" "5/3/2016 11:59:59 PM" "5/2/2016 11:59:59 P
##  $ WeightKg                : num  52.6 52.6 52.6 52.6 52.6 ...
##  $ WeightPounds            : num  116 116 116 116 116 ...
##  $ Fat                     : int  22 NA 22 NA 22 NA 22 NA 22 NA ...
##  $ BMI                     : num  22.6 22.6 22.6 22.6 22.6 ...
##  $ IsManualReport          : chr  "True" "True" "True" "True" ...
##  $ LogId                   : num  1.46e+12 1.46e+12 1.46e+12 1.46e+12 1.46e+12 ...
```

```r
nrow(all_tables)
```

```
## [1] 1339046
```

```r
set.seed(123)
dat.d <- sample(1:nrow(all_tables),size=nrow(all_tables)*0.7,replace = FALSE) #random selection of 70%

train.loan <- all_tables[dat.d,] # 70% training data
test.loan <- all_tables[-dat.d,] # remaining 30% test data
```

Limitations :

Data was collected in 2016, hence data may not be relevant to modern trends. Small sample size of only 30 participants. Data does not include demographics about the sample such as sex, age, or geographical location. This may not be a good representation of the population of women globally who would use a similar product.Survey style of data collection may be subject to response bias. Integrity and accuracy of data is not clear.

Inital observations of these CSVs within Mircosoft Excel shows that these files contain acitvites, calory records, physical acitivity records, step record, sleep monitoring, heart rate, weight and BMI calculations. Using simple unique formula against unique ID of users bring out the fact that these files contain the above mentioned data for anywhere between 8 to 33 users. Another point to be noted here is the fact that some of these numbers are manual input of users, such as weight in the weightLogInfo_merged.csv file.

Concluding Remarks =================== :

Of course, we need much more data to draw conclusions but this preliminary analysis looks promising and suggests that there may be a correlation between Tracker Distance,Calories and Weight Pounds.More tracking is happened by wearable devices that can help you in archiving your goals and finally make sense of all the data generated by your watch.