Bellabeat Case Study

I have already used SQL and pivot tables in spreadsheets to do some initial analysis. I want to rely on R programming language to help create visualizations of the data. I will begin by loading the appropriate packages.

```
install.packages("tidyverse")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```
library(tidyverse)
```

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

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

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

```
library(readr)
```

I will now upload my datasets. Because I was able to combine the hourly data in spreadsheets, I can upload that on its own, but I am going to have to rely on joins to aggregate the sleep data into the daily dataset.

```
dailyactivity <- read_csv("Bellabeat_Case_Study/Bellabeat_daily.csv")
```

```
## Rows: 940 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ActivityDate
## dbl (6): Id, TotalSteps, TotalDistance, TotalActiveMinutes, SedentaryMinutes...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dailysleep <- read_csv("Bellabeat_Case_Study/Bellabeat_sleep.csv")
```

```
## Rows: 413 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): SleepDate
## dbl (3): Id, TotalMinutesAsleep, TotalTimeInBed
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
hourly <- read_csv("Bellabeat_Case_Study/Bellabeat_hourly.csv")
```

```
## Rows: 22099 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Activity Day
## dbl  (4): Id, Calories, Total Intensities, Steps
## time (1): Activity Hour
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
weekdaysleep <- read_csv("Bellabeat_Case_Study/Bellabeat_avgSleepDay.csv")
```

```
## Rows: 504 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): Weekday, ActivityDate, Avg_Distance
## dbl (4): Avg_Sedentary, Avg_Sleep, Avg_Steps, Avg_Calories
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
weekday <- read_csv("Bellabeat_Case_Study/Bellabeat_weekday.csv")
```

```
## Rows: 7 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Weekday
## dbl (4): Sedentary, Sleep, Steps, Calories
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(dailyactivity)
```

```
## # A tibble: 6 x 7
##       Id ActivityDate TotalSteps TotalDistance TotalActiveMinu~ SedentaryMinutes
##    <dbl> <chr>             <dbl>         <dbl>            <dbl>            <dbl>
## 1 1.50e9 4/12/2016         13162          8.5               366              728
## 2 1.50e9 4/13/2016         10735          6.97              257              776
## 3 1.50e9 4/14/2016         10460          6.74              222             1218
## 4 1.50e9 4/15/2016          9762          6.28              272              726
## 5 1.50e9 4/16/2016         12669          8.16              267              773
## 6 1.50e9 4/17/2016          9705          6.48              222              539
## # ... with 1 more variable: Calories <dbl>
```

```
head(dailysleep)
```

```
## # A tibble: 6 x 4
##           Id SleepDate TotalMinutesAsleep TotalTimeInBed
##        <dbl> <chr>                  <dbl>          <dbl>
## 1 1503960366 4/12/2016                327            346
## 2 1503960366 4/13/2016                384            407
## 3 1503960366 4/15/2016                412            442
## 4 1503960366 4/16/2016                340            367
## 5 1503960366 4/17/2016                700            712
## 6 1503960366 4/19/2016                304            320
```

```r
head(hourly)
```

```
## # A tibble: 6 x 6
##           Id `Activity Day` `Activity Hour` Calories `Total Intensities` Steps
##        <dbl> <chr>          <time>            <dbl>               <dbl> <dbl>
## 1 1503960366 4/12/2016      00:00                81                  20   373
## 2 1503960366 4/12/2016      01:00                61                   8   160
## 3 1503960366 4/12/2016      02:00                59                   7   151
## 4 1503960366 4/12/2016      03:00                47                   0     0
## 5 1503960366 4/12/2016      04:00                48                   0     0
## 6 1503960366 4/12/2016      05:00                48                   0     0
```

```r
head(weekdaysleep)
```

```
## # A tibble: 6 x 7
##   Weekday  ActivityDate Avg_Sedentary Avg_Sleep Avg_Steps Avg_Calories
##   <chr>    <chr>                <dbl>     <dbl>     <dbl>        <dbl>
## 1 Thursday 5/5/2016              749.      362.    10255.        2550.
## 2 Tuesday  4/26/2016             780.      369.     9290.        2444.
## 3 Thursday 4/21/2016             791.      376      9698.        2579.
## 4 Friday   4/29/2016             716.      386.     7910.        2269.
## 5 Saturday 4/16/2016             710.      392.     8615.        2494.
## 6 Friday   4/22/2016             739.      393.     8377.        2456.
## # ... with 1 more variable: Avg_Distance <chr>
```

```r
head(weekday)
```

```
## # A tibble: 6 x 5
##   Weekday  Sedentary Sleep Steps Calories
##   <chr>        <dbl> <dbl> <dbl>    <dbl>
## 1 Friday         741   405  7936     2332
## 2 Monday         718   419  9262     2435
## 3 Saturday       682   418  9857     2504
## 4 Sunday         686   455  7295     2279
## 5 Thursday       660   405  7810     2208
## 6 Tuesday        740   405  9183     2499
```

And now to join the daily data.

```r
daily_join <- right_join(dailyactivity, dailysleep, by = c('Id'='Id','ActivityDate'='SleepDate'))
head(daily_join)
```

```
## # A tibble: 6 x 9
##       Id ActivityDate TotalSteps TotalDistance TotalActiveMinu~ SedentaryMinutes
##    <dbl> <chr>             <dbl>         <dbl>            <dbl>            <dbl>
## 1 1.50e9 4/12/2016         13162          8.5               366              728
## 2 1.50e9 4/13/2016         10735          6.97              257              776
## 3 1.50e9 4/15/2016          9762          6.28              272              726
## 4 1.50e9 4/16/2016         12669          8.16              267              773
## 5 1.50e9 4/17/2016          9705          6.48              222              539
## 6 1.50e9 4/19/2016         15506          9.88              345              775
## # ... with 3 more variables: Calories <dbl>, TotalMinutesAsleep <dbl>,
## #   TotalTimeInBed <dbl>
```

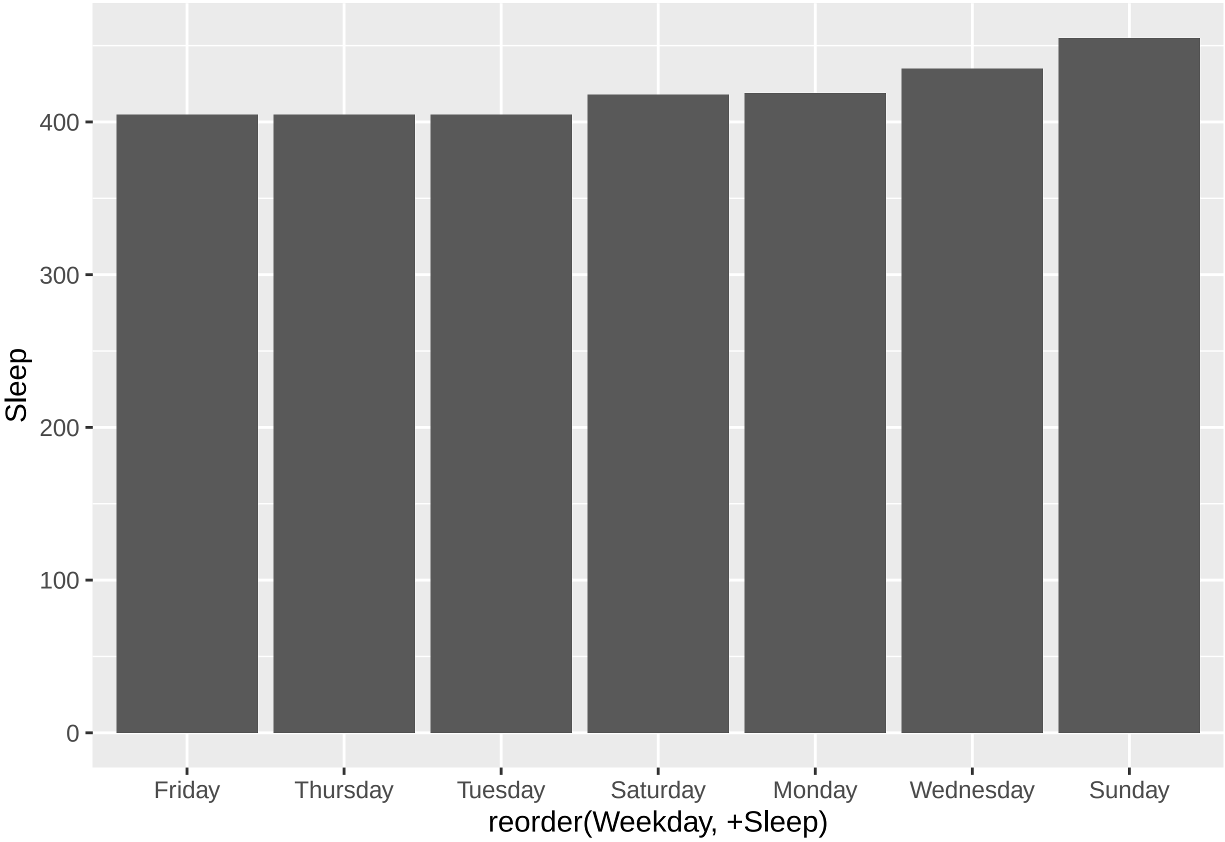
Now we have a complete, clean dataset for the daily data. We are now going to try out some different visualizations.

The first question became, "What do we qualify as a healthy user?" After some research, these are the following metrics we are going to use *Even though gender is not specified, I'm going to use suggested metrics related to women* - Recommended for adults to get between 7-9 hours of sleep a night - Recommended for adults to get around 10,000 steps a day - Recommended for adult women to burn around 2,000 calories a day

I will use these to qualify a "healthy" user.

Using spreadsheets and SQL I have already discovered: - 33 users provided data - 24 users provided sleep data - 413 individual sleeps recorded - Only 119 sleep days that fall between 7-9 hours - 19 out of the 24 users averaged suggested sleep over the 30 days

```
ggplot(weekday, mapping = aes(x=reorder(Weekday,+Sleep), y=Sleep)) + geom_col()
```

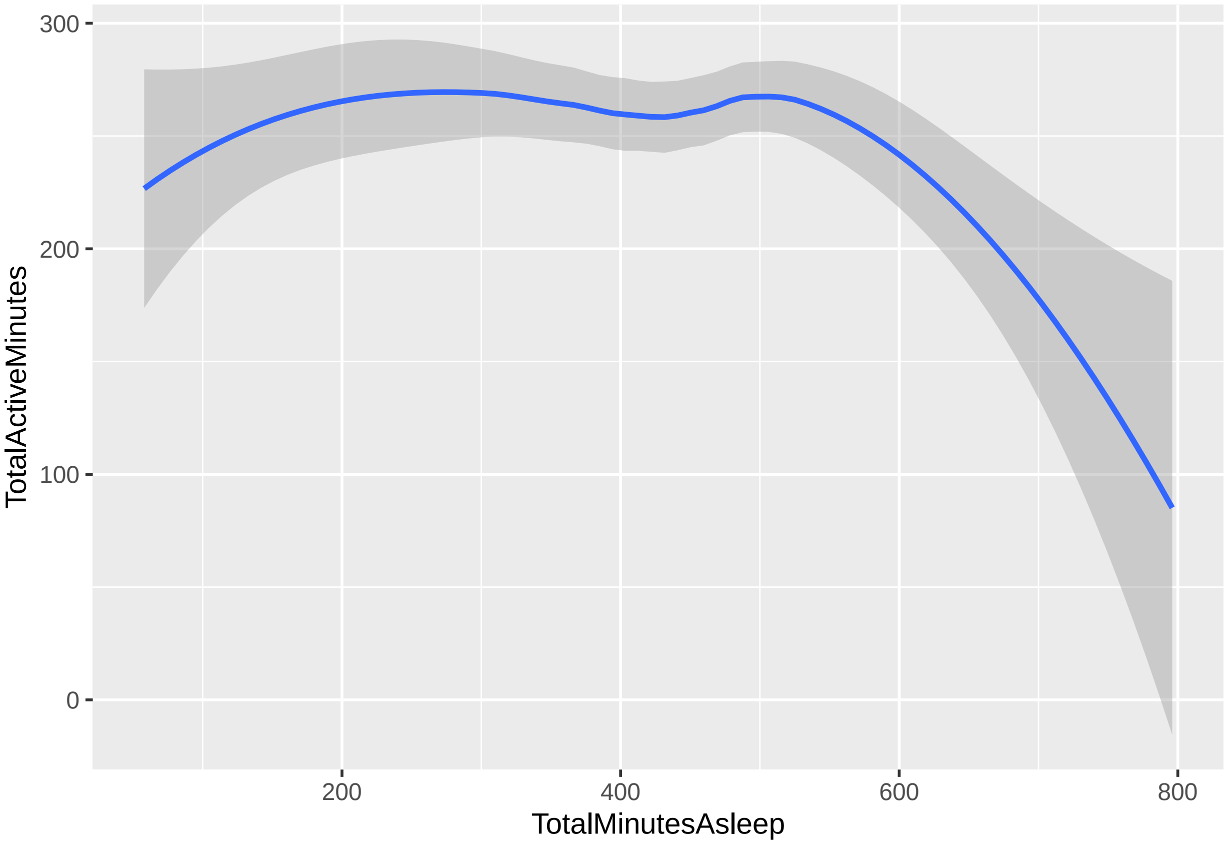


This first visualization shows the average sleep grouped by weekday. So on average, Fridays, Thursdays, and Tuesdays get the least amount of sleep.

So the next question is how do we help users get more sleep? My hypothesis is that increased activity leads to increased sleep.

```
ggplot(daily_join, mapping = aes(x=TotalMinutesAsleep,y=TotalActiveMinutes)) + geom_smooth()
```

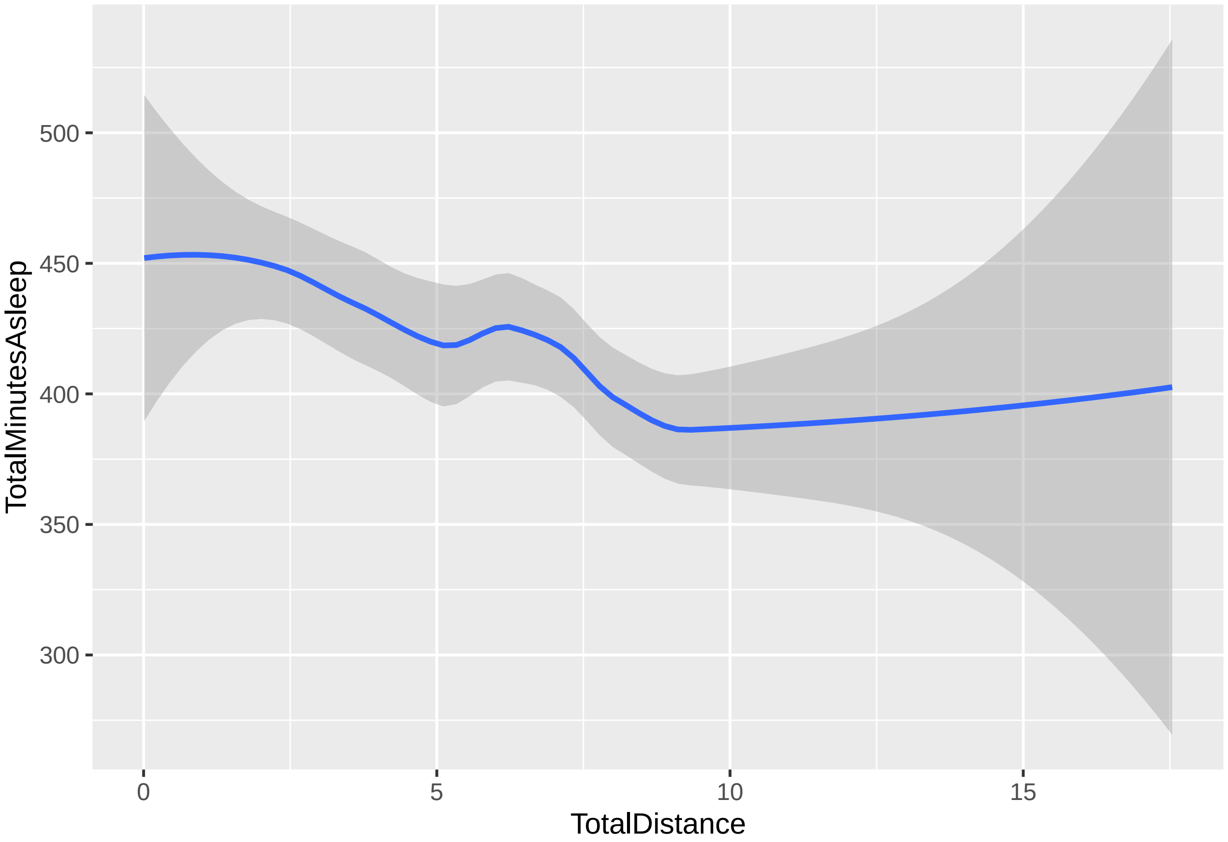
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



There actually isn't a real correlation between the two. So using my remaining data points, I'm going to check for correlation between them

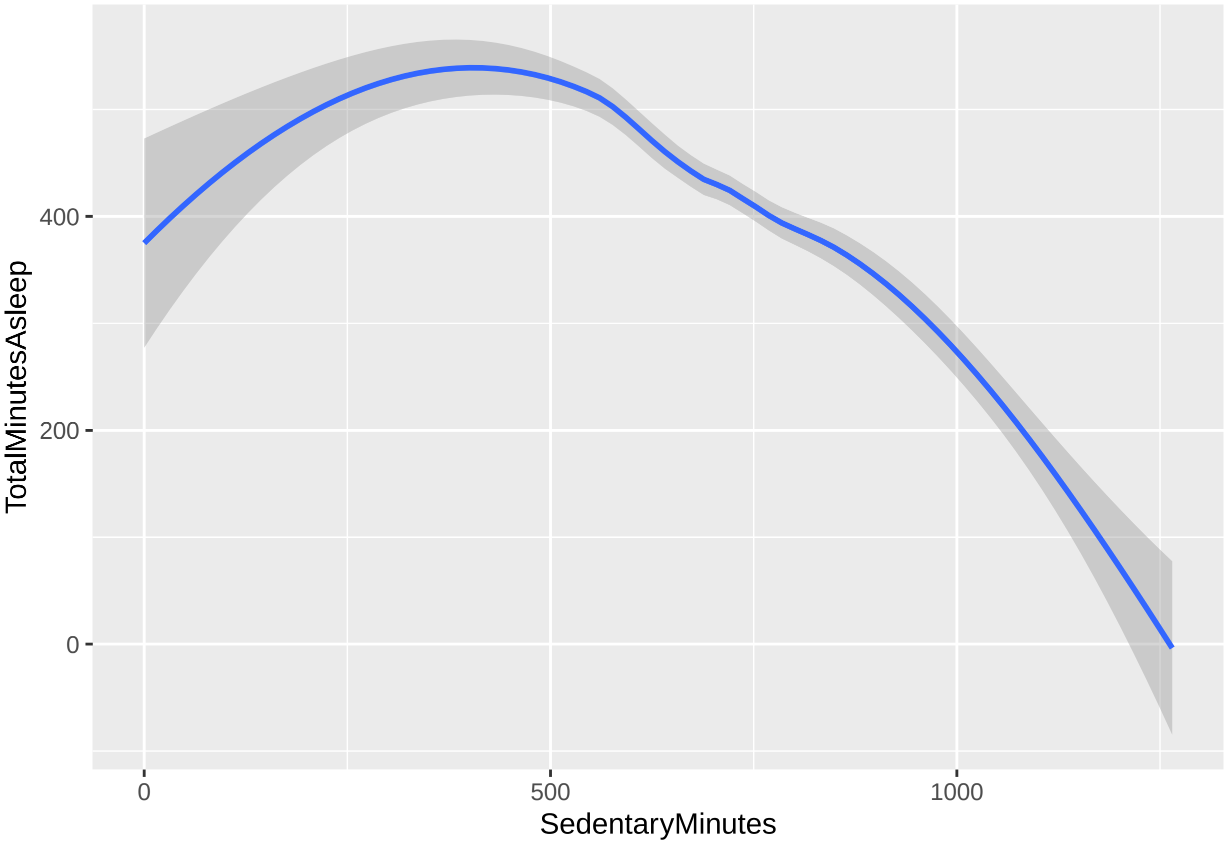
```
ggplot(daily_join, mapping = aes(x=TotalDistance,y=TotalMinutesAsleep)) + geom_smooth()
```

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



```
ggplot(daily_join, mapping = aes(x=SedentaryMinutes,y=TotalMinutesAsleep)) + geom_smooth()
```

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



Of the three, the strongest negative correlation is between sedentary minutes and total minutes asleep. Let me try to look at this from another angle.

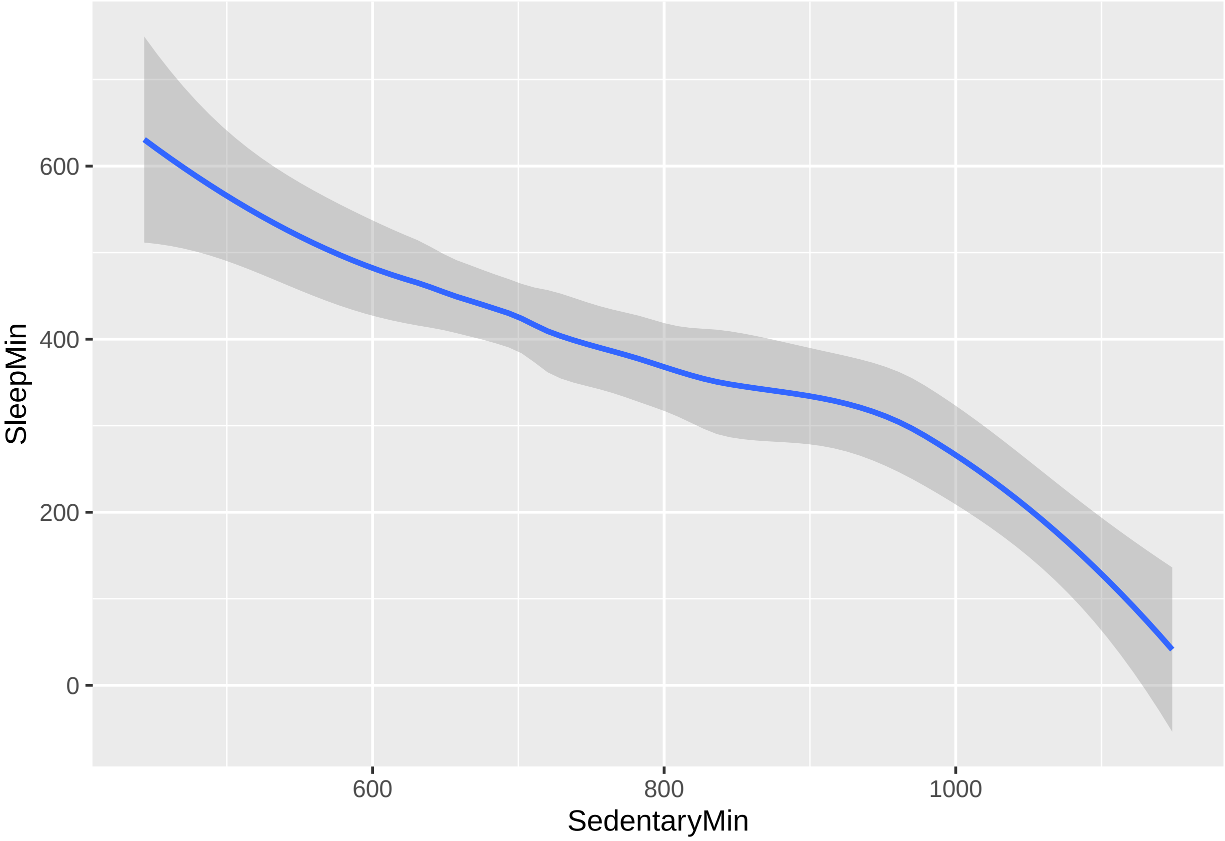
```
AvgId <- aggregate(cbind(Active <- daily_join$TotalActiveMinutes, Sedentary <- daily_join$SedentaryMinu
AvgId <- setNames(AvgId, c("Id", "ActivityMin", "SedentaryMin", "Steps", "SleepMin"))
arrange(AvgId, desc(SleepMin), .by_group = FALSE)
```

```
##             Id ActivityMin SedentaryMin     Steps SleepMin
## 1  1844505072    147.3333     443.3333  3477.000 652.0000
## 2  2026352035    256.8929     653.9643  5618.679 506.1786
## 3  6117666160    364.5556     531.9444  8823.833 478.7778
## 4  4319703577    259.2308     642.6923  7125.423 476.6538
## 5  5553957443    242.6129     668.3548  8612.581 463.4839
## 6  7086361926    234.0000     723.6667 10290.500 453.1250
## 7  6962181067    287.1290     662.3226  9794.806 448.0000
## 8  2347167796    271.2000     628.4000  8533.200 446.8000
## 9  8378563200    226.5625     715.3750  8832.938 443.3438
## 10 8792009665    178.4000     807.8000  3443.267 435.6667
## 11 5577150313    296.2308     667.3077  9260.077 432.0000
## 12 4702921684    292.1429     693.0357  9226.357 421.1429
## 13 1927972279     85.0000     977.2000  1490.000 417.0000
## 14 4388161847    286.7500     751.4583 10974.708 403.1250
## 15 4445114986    217.9643     787.3214  4756.179 385.1786
## 16 1503960366    291.3200     759.2800 12405.680 360.2800
## 17 6775888955    107.0000     964.0000  3499.000 349.6667
## 18 4020332650    249.0000     841.8750  6596.750 349.3750
## 19 8053475328    301.0000     837.3333 19078.667 297.0000
## 20 1644430081    263.2500     920.5000  7967.750 294.0000
## 21 3977333714    262.6429     716.2143 11218.000 293.6429
## 22 4558609924    313.0000    1028.4000  8139.000 127.6000
## 23 7007744171    220.0000    1148.5000  5115.500  68.5000
## 24 2320127002    242.0000    1129.0000  5079.000  61.0000
```

That gave me a tibble of the basic averages per each user with data on activity minutes, sedentary minutes, steps, and sleep minutes. I now want to see if the averages have any more correlation.

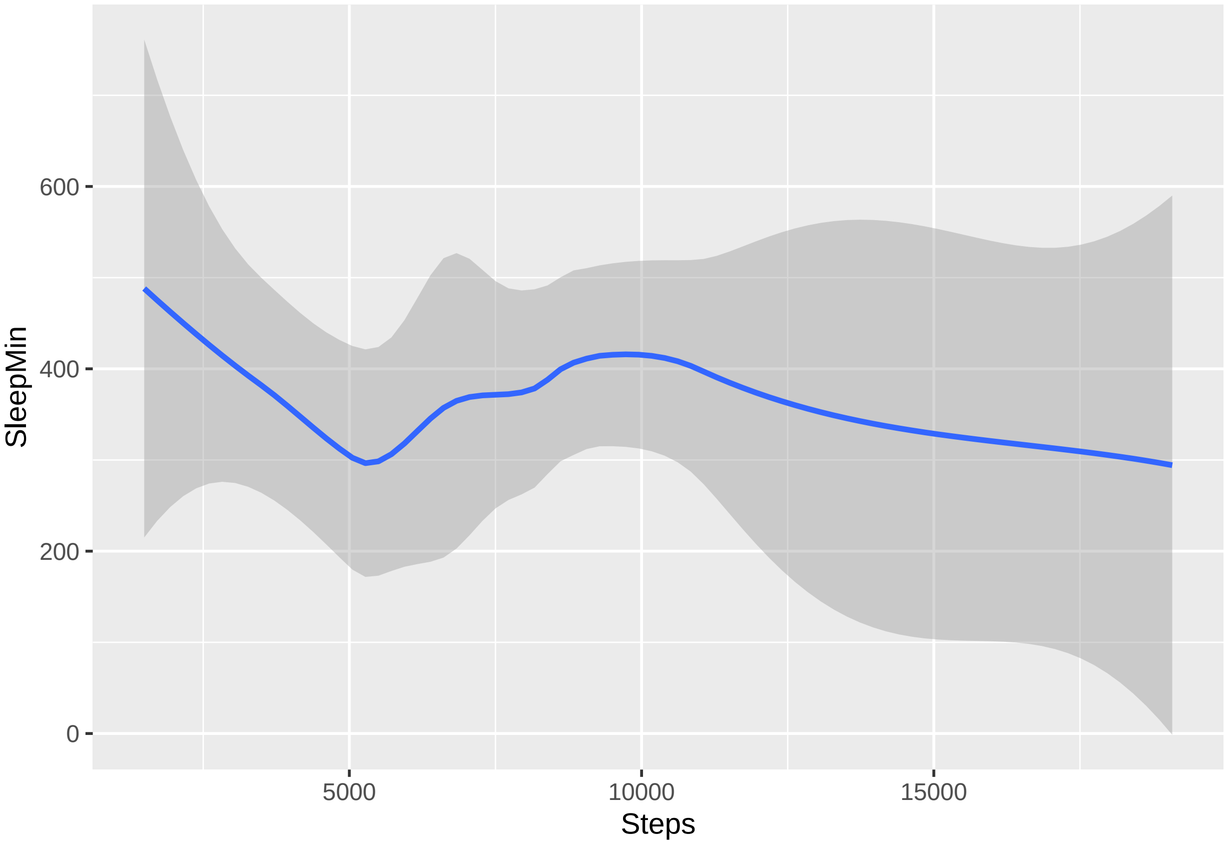
```
ggplot(AvgId, mapping = aes(x=SedentaryMin, y=SleepMin)) + geom_smooth()
```

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



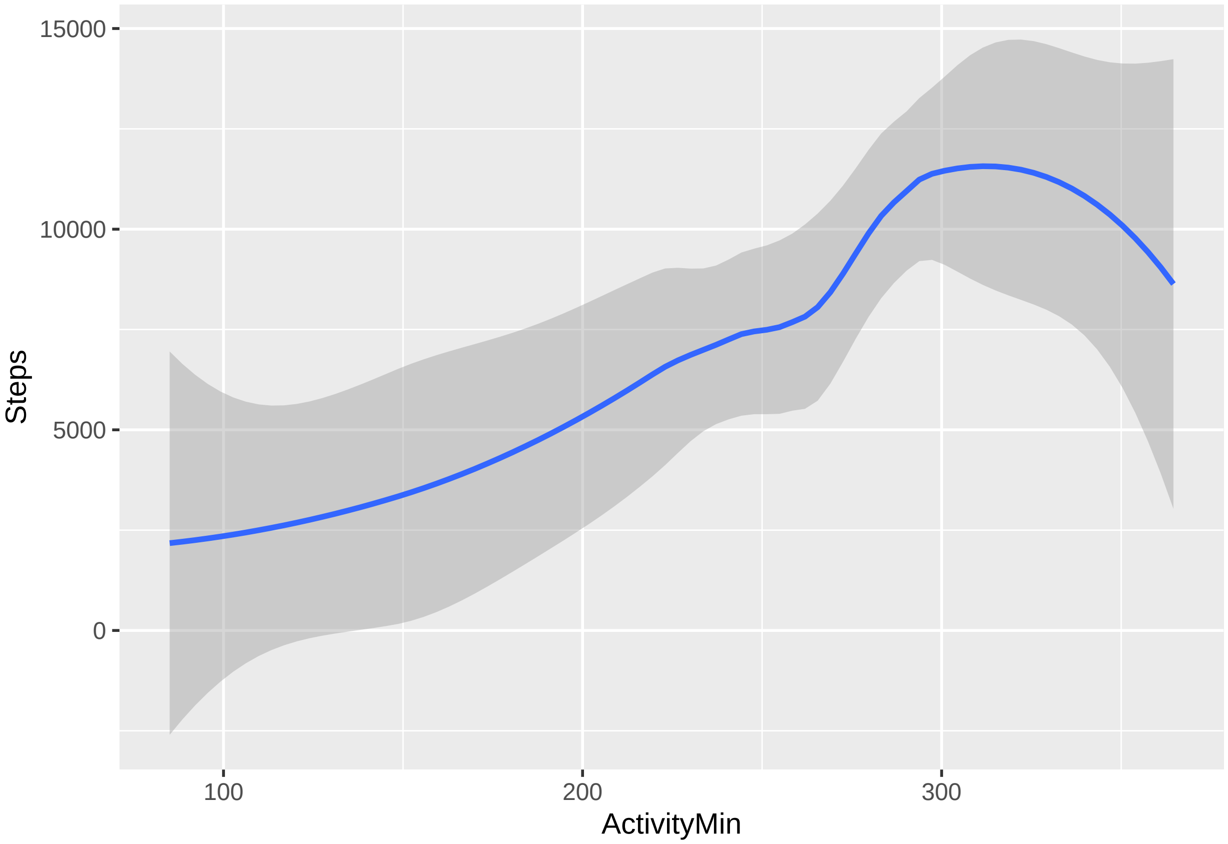
```
ggplot(AvgId, mapping = aes(x=Steps, y=SleepMin)) + geom_smooth()
```

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



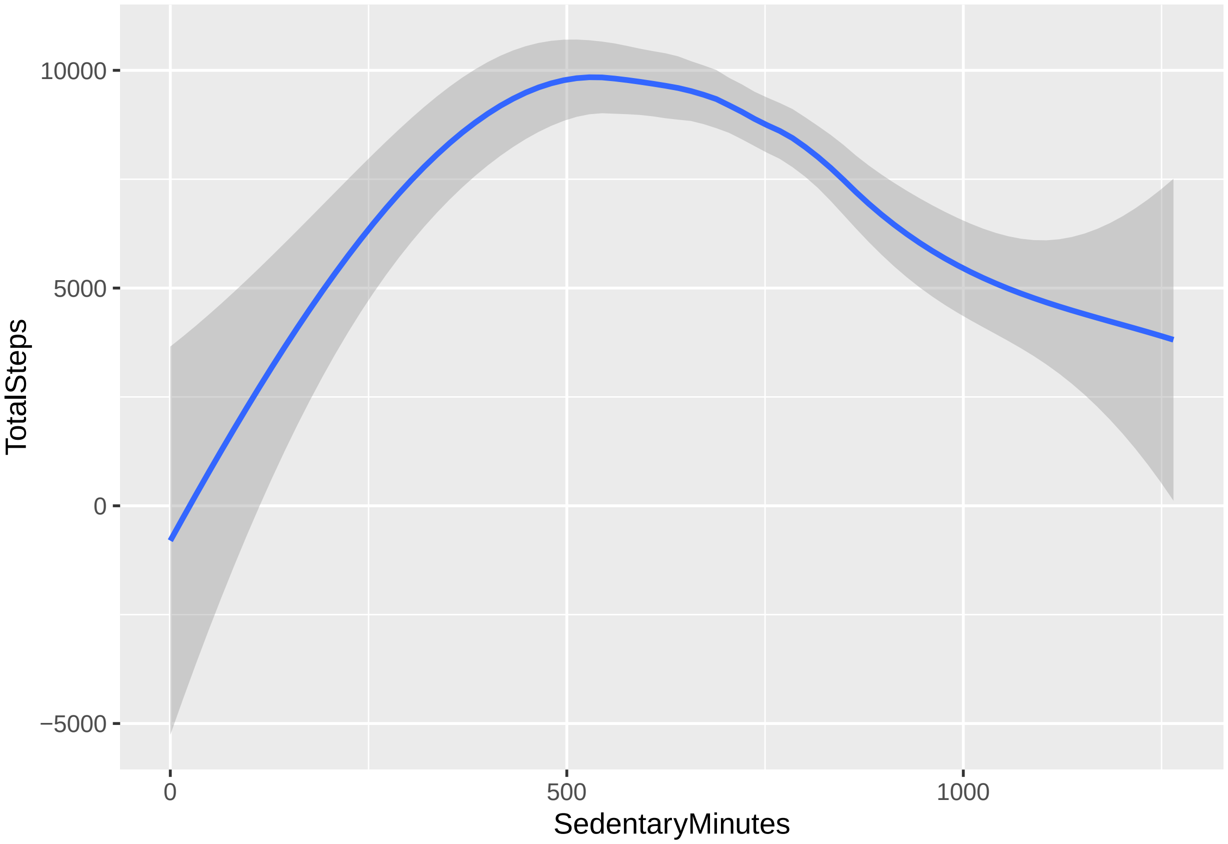
```
ggplot(AvgId, mapping = aes(x=ActivityMin, y=Steps)) + geom_smooth()
```

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



```
ggplot(daily_join, mapping = aes(x=SedentaryMinutes,y=TotalSteps)) +geom_smooth()
```

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



Even using a different set of averages, it still shows a negative correlation between time spent sedentary and time spent asleep. Rather than increasing exercise, Bellabeat should focus on reducing sedentary time to increase sleep.

```
hourly_mod <- aggregate(cbind(hourly$Calories, as.integer(hourly$`Total Intensities`), hourly$Steps), li
setNames(hourly_mod, c("ActivityHour", "Calories", "Intensity", "Steps"))
```

```
##    ActivityHour  Calories  Intensity       Steps
## 1      00:00:00  71.80514  7.3158458  140.389722
## 2      01:00:00  70.16506  3.5069668   50.526259
## 3      02:00:00  69.18650  1.8510182   31.326902
## 4      03:00:00  67.53805  0.9721329   11.675241
## 5      04:00:00  68.26180  0.3798283    3.589056
## 6      05:00:00  81.70815  0.1673820    2.115880
## 7      06:00:00  86.99678  0.8979592   13.676692
## 8      07:00:00  94.47798  2.5273899   38.126745
## 9      08:00:00 103.33727  8.8818475  185.514501
## 10     09:00:00 106.14286 22.4167562 1063.537057
## 11     10:00:00 110.46071 15.9149623  443.417653
## 12     11:00:00 109.80690 14.1974110  462.752967
## 13     12:00:00 117.19740 18.4446855  559.980477
## 14     13:00:00 115.30945 21.2931596  708.247557
## 15     14:00:00 115.73290 23.2877307  737.046688
## 16     15:00:00 106.63716 14.5890710  450.953005
## 17     16:00:00 113.32745 18.9834620  568.424476
## 18     17:00:00 122.75276 20.8410596  698.151214
## 19     18:00:00 123.49227 43.6545254 1520.247241
## 20     19:00:00 121.48455 31.5673289  984.312362
## 21     20:00:00 102.35762 31.6004415 1085.163355
## 22     21:00:00  96.05635 37.0099448 1301.507182
## 23     22:00:00  88.26549 20.8440265  656.394912
## 24     23:00:00  77.59358 24.0365449  745.544850
```

```
arrange(hourly_mod, desc("Intensity"), .by_group = TRUE)
```

```
##     Group.1        V1         V2          V3
## 1  00:00:00  71.80514  7.3158458  140.389722
## 2  01:00:00  70.16506  3.5069668   50.526259
## 3  02:00:00  69.18650  1.8510182   31.326902
## 4  03:00:00  67.53805  0.9721329   11.675241
## 5  04:00:00  68.26180  0.3798283    3.589056
## 6  05:00:00  81.70815  0.1673820    2.115880
## 7  06:00:00  86.99678  0.8979592   13.676692
## 8  07:00:00  94.47798  2.5273899   38.126745
## 9  08:00:00 103.33727  8.8818475  185.514501
## 10 09:00:00 106.14286 22.4167562 1063.537057
## 11 10:00:00 110.46071 15.9149623  443.417653
## 12 11:00:00 109.80690 14.1974110  462.752967
## 13 12:00:00 117.19740 18.4446855  559.980477
## 14 13:00:00 115.30945 21.2931596  708.247557
## 15 14:00:00 115.73290 23.2877307  737.046688
## 16 15:00:00 106.63716 14.5890710  450.953005
## 17 16:00:00 113.32745 18.9834620  568.424476
## 18 17:00:00 122.75276 20.8410596  698.151214
## 19 18:00:00 123.49227 43.6545254 1520.247241
## 20 19:00:00 121.48455 31.5673289  984.312362
## 21 20:00:00 102.35762 31.6004415 1085.163355
## 22 21:00:00  96.05635 37.0099448 1301.507182
## 23 22:00:00  88.26549 20.8440265  656.394912
## 24 23:00:00  77.59358 24.0365449  745.544850
```

So this now breaks down average calories, intensities, and steps based on hour of the day. Although I had issues arranging the data set, you can still see that (excluding normal sleeping hours) 5AM-8AM have the lowest intensities recorded

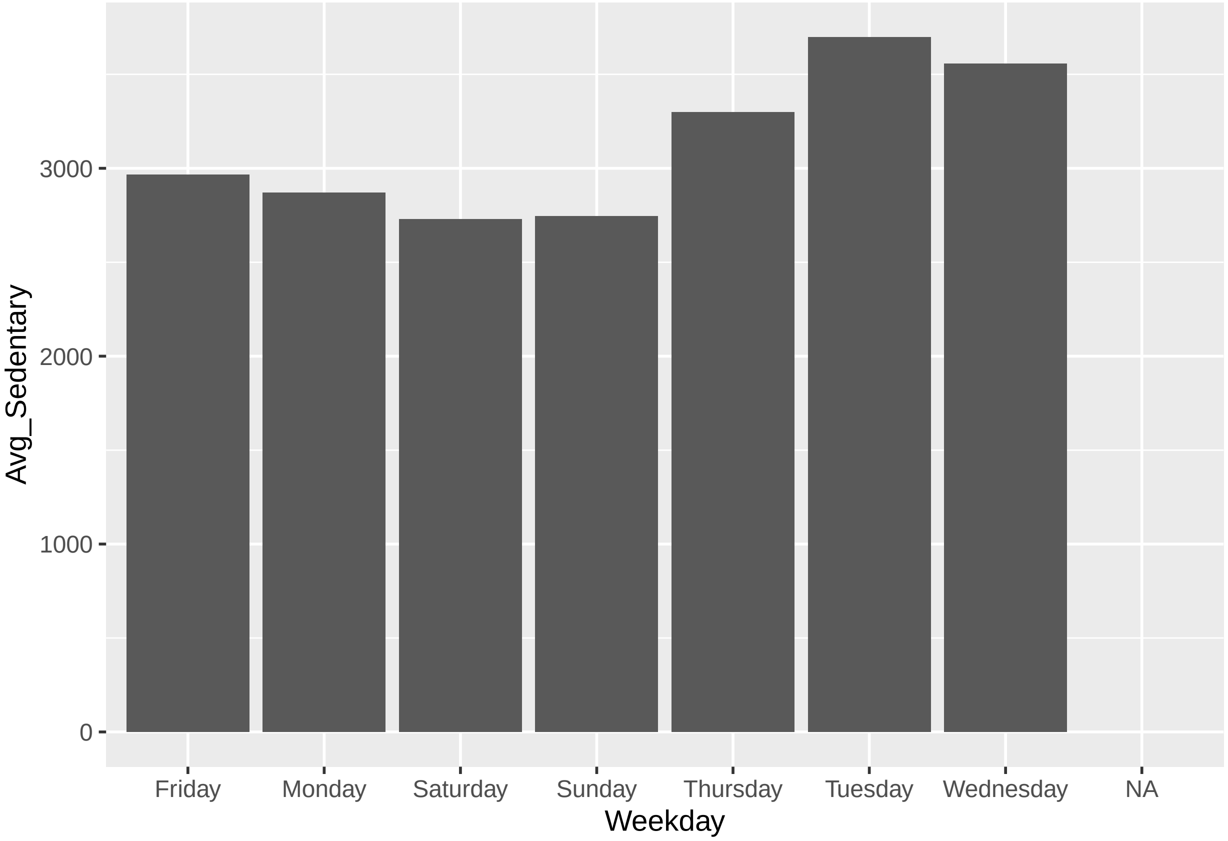
```
weekdaysleep %>%
  group_by(Weekday)%>%
  summarise(sedentary = mean(Avg_Sedentary, na.rm=FALSE), sleep = mean(Avg_Sleep, na.rm=FALSE))
```

```
## # A tibble: 8 x 3
##   Weekday   sedentary sleep
##   <chr>         <dbl> <dbl>
## 1 Friday         741.  405.
## 2 Monday         718.  419.
## 3 Saturday       682.  418.
## 4 Sunday         686.  455.
## 5 Thursday       660.  405.
## 6 Tuesday        740.  405.
## 7 Wednesday      712.  435.
## 8 <NA>            NA    NA
```

This breaks down daily averages of sedentary time and sleep time.

```
ggplot(weekdaysleep, mapping = aes(x=Weekday, y=Avg_Sedentary)) + geom_col()
```

```
## Warning: Removed 473 rows containing missing values (position_stack).
```



As shown here, Tuesday-Thursday have the highest average sedementary time.