Bellabeat Case Study

Cyris Zeiders

2023-05-01

INTRODUCTION

Hello World, this is my rendition of the Bellabeat Case Study brought to you by Coursera's Google Data Analytics Course.

SITUATION

In my made-up world, my company is seeking to use acquired fitbit data to analyze which days of the week our clients are most active, as well as what days they sleep the best.

A secondary goal is to also find those same metrics, but while highlighting the difference in results according to whether the user is considered an "All Day Wearer" i.e. wears the device for longer than twelve hours.

SOURCE DATA

- Kaggle - Bellabeat

R CODE

INSTALL PACKAGES

Here, I install 'tidyverse', 'lubridate', and 'janitor' for the ease of cleaning provided data.

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

RUN PACKAGES

Unpack the contents of the packages.

```
library(tidyverse)
library(lubridate)
library(janitor)
library(scales)
```

IMPORT CSV

Import the supplied data and assign each CSV into separate data frames.

```
activity_df <- read_csv("dailyActivity_merged.csv")
sleep_df <- read_csv("sleepDay_merged.csv")
```

CLEANING 'activity_df'

Create 'trim_activity_df' to hold trimmed data from 'activity_df'.

Step one consisted of:

- Dropping NAs
- Mutating 'Very', 'Fairly', and 'Lightly Active Minutes' into new variable 'ActiveMinutes'
- Select dropping unused columns

```
trim_activity_df <- activity_df %>%
  drop_na() %>%
  mutate(ActiveMinutes =
           VeryActiveMinutes +
           FairlyActiveMinutes +
           LightlyActiveMinutes) %>%
  select(-c(TrackerDistance,
       LoggedActivitiesDistance,
       VeryActiveDistance,
       ModeratelyActiveDistance,
       LightActiveDistance,
       SedentaryActiveDistance,
       VeryActiveMinutes,
       FairlyActiveMinutes,
       LightlyActiveMinutes))
```

Step two consisted of changing 'ActivityDate' from characters to datetime.

```
trim_activity_df$ActivityDate <- mdy(trim_activity_df$ActivityDate)
```

CLEANING 'sleep_df'

Created 'trim_sleep_df' to hold trimmed data from 'sleep_df'.

Step one consisted of:

- Dropping NAs
- Mutating 'TotalTimeInBed' and 'TotalMinutesAsleep' into new variable 'TimeAwakeInBed'
- Select dropping unused columns

```
trim_sleep_df <- sleep_df %>%
  drop_na() %>%
  mutate(TimeAwakeInBed =
           TotalTimeInBed -
           TotalMinutesAsleep) %>%
  select(-TotalSleepRecords)
```

Step two consisted of:

- Assigning 'SleepDay' column the name 'ActivityDate' to match 'trim_activity_df'
- Dropping the time portion of the 'ActivityDate'
- Change 'ActivityDate' from characters to datetime

```
colnames(trim_sleep_df)[2] <- "ActivityDate"

trim_sleep_df$ActivityDate <- format(mdy_hms(trim_sleep_df$ActivityDate),
                                     format = "%m/%d/%Y")

trim_sleep_df$ActivityDate <- mdy(trim_sleep_df$ActivityDate)
```

COMBINE BOTH DATA FRAMES

Create 'data_df' to hold combined data from both 'trim_activity_df' and 'trim_sleep_df'.

Next, INNER JOIN both data frames by 'Id' and 'ActivityDate'.

```
data_df <- trim_activity_df %>%
  inner_join(trim_sleep_df, by = c("Id", "ActivityDate"))
```

CLEANING 'data_df'

Rename columns so they retain original name, but are lowercased.

```
names(data_df) <- tolower(names(data_df))
```

Next,

- Separate the month from date into a separate variable
- Separate the weekday from the date into a separate variable

```
data_df$month <- months(ymd(data_df$activitydate))

data_df$weekday <- weekdays(ymd(data_df$activitydate))
```

Then,

- Convert zeros in NAs
- Remove NAs from data frame

```
data_df[data_df == 0] <- NA

data_df <- data_df[complete.cases(data_df),]
```

CRUNCHING NUMBERS

Find the total amount of minute the device was work by combining 'sedentary' and 'active minutes', then store as new column.

```
data_df$minutesworn <- (data_df$sedentaryminutes + data_df$activeminutes)
```

Assign a TRUE or FALSE in a new column by determining if newly created 'minutesworn' column is equal to or greater than 720 minutes, or twelve hours.

```
data_df$alldaywear <- data_df$minutesworn >= 720
```

PLOTTING BY 'weekday'

AVERAGE ACTIVE MINUTES BY DAY

First, find the MAX of the MEAN for each 'weekday'.

```
data_df %>%  group_by(weekday) %>% summarize(max(mean(activeminutes)))
```

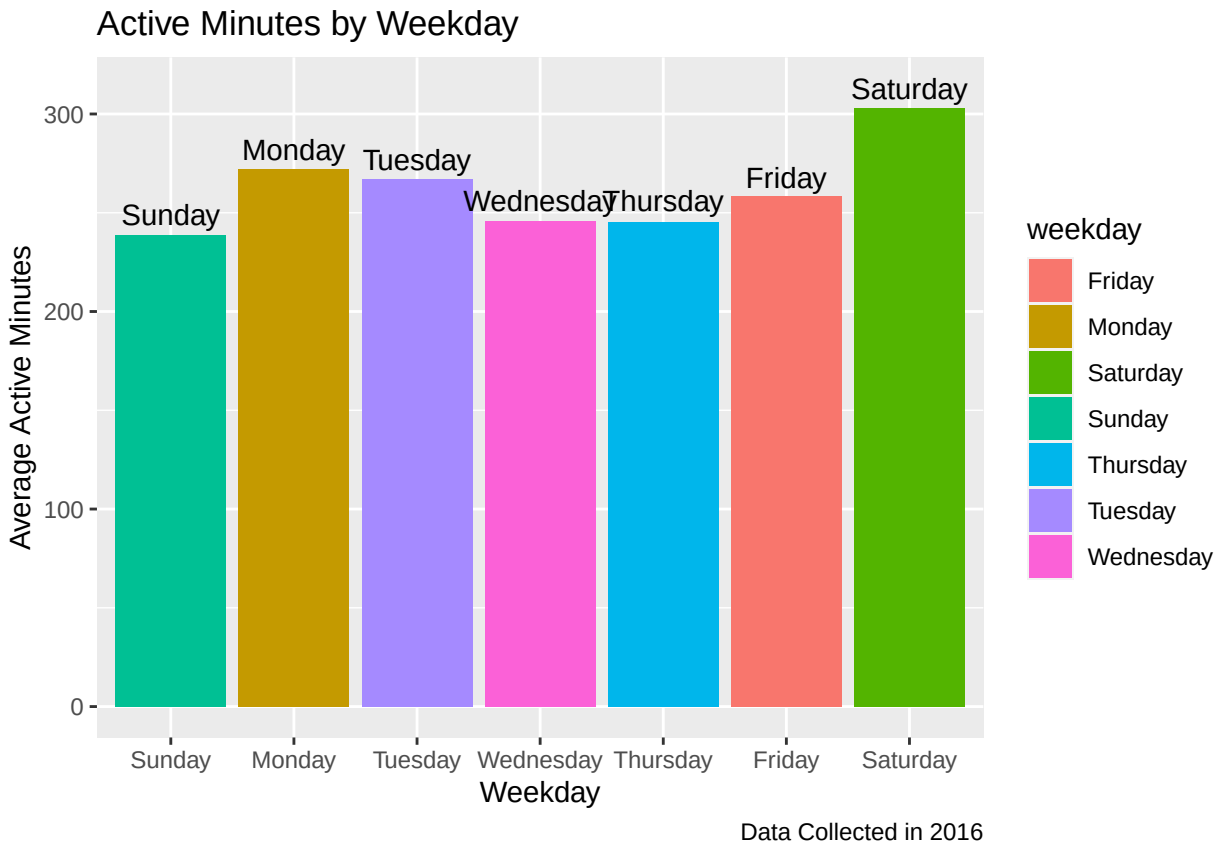
```
## # A tibble: 7 x 2
##   weekday   `max(mean(activeminutes))`
##   <chr>                          <dbl>
## 1 Friday                          258.
## 2 Monday                          272.
## 3 Saturday                        303.
## 4 Sunday                          239.
```

```
## 5 Thursday                      246.
## 6 Tuesday                       267.
## 7 Wednesday                     246.
```

Next,

- Group by 'weekday'
- Summarize the MEAN of 'activeminutes' into new variable 'mean_active'
- Begin plotting by organizing x-axis column names into calendar format
- Assign 'mean_active' to y-axis
- Fill according to 'weekday'
- Assign 'geom_col'
- Add title and caption
- Add X and Y axis labels
- Assign appropriate annotation to each 'weekday'

```
data_df %>%
  group_by(weekday) %>%
  summarize(mean_active = mean(activeminutes)) %>%
  ggplot(mapping = aes(x = factor(weekday, level =
                                    c('Sunday', 'Monday', 'Tuesday',
                                      'Wednesday', 'Thursday', 'Friday',
                                      'Saturday')),
                       y = mean_active, fill = weekday)) +
  geom_col() +
  labs(title = "Active Minutes by Weekday",
       caption = "Data Collected in 2016") +
  xlab("Weekday") + ylab("Average Active Minutes") +
  annotate("text", x = "Friday", y = 268, label = "Friday") +
  annotate("text", x = "Saturday", y = 313, label = "Saturday") +
  annotate("text", x = "Sunday", y = 249, label = "Sunday") +
  annotate("text", x = "Monday", y = 282, label = "Monday") +
  annotate("text", x = "Tuesday", y = 277, label = "Tuesday") +
  annotate("text", x = "Wednesday", y = 256, label = "Wednesday") +
  annotate("text", x = "Thursday", y = 256, label = "Thursday")
```



According to the graph;

- Saturday is MOST active
- Sunday is LEAST active

AVERAGE CALORIES BURNED BY DAY

First, find the MAX of the MEAN for each 'weekday'.

```
data_df %>%  group_by(weekday) %>% summarize(max(mean(calories)))
```

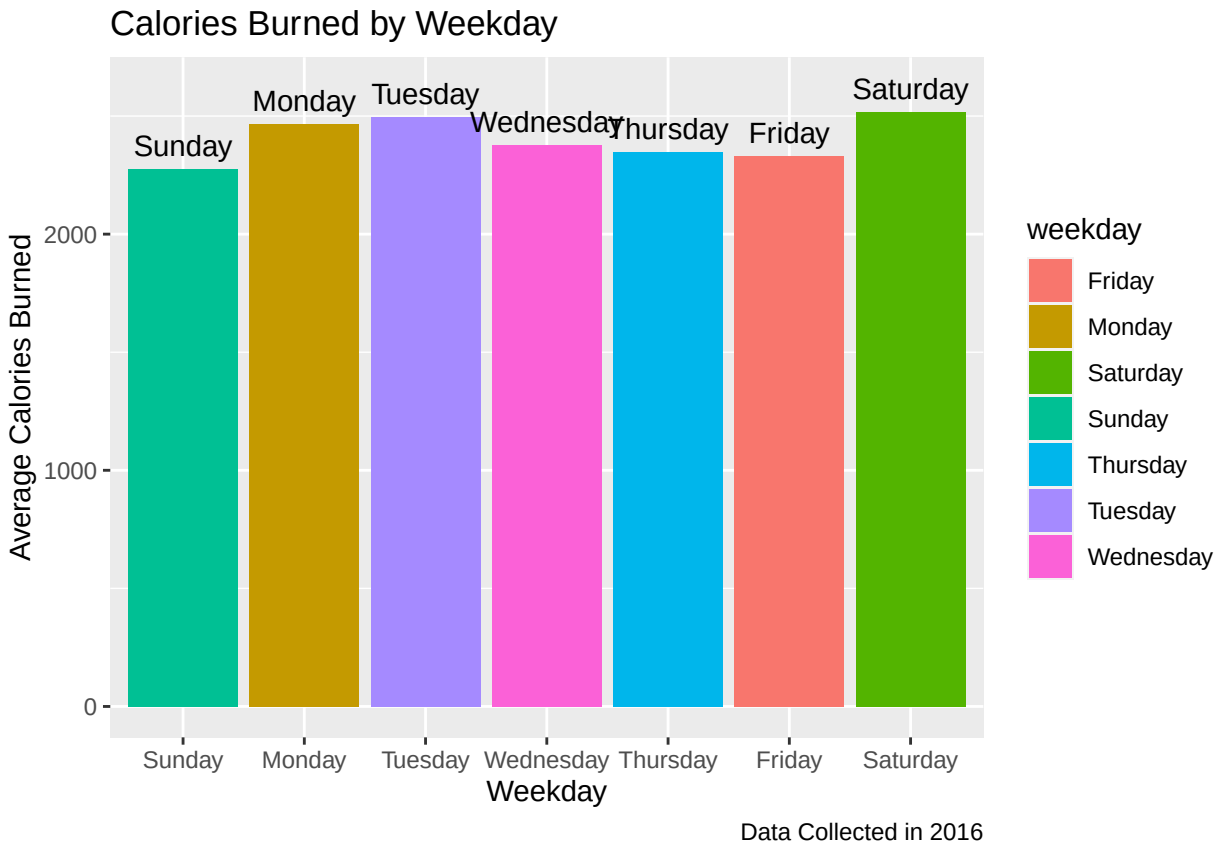
```
## # A tibble: 7 x 2
##   weekday   `max(mean(calories))`
##   <chr>                     <dbl>
## 1 Friday                    2330.
## 2 Monday                    2465.
## 3 Saturday                  2518.
## 4 Sunday                    2277.
## 5 Thursday                  2348.
## 6 Tuesday                   2496.
## 7 Wednesday                 2378.
```

Next,

- Group by 'weekday'
- Summarize the MEAN of 'calories' into new variable 'mean_calories'
- Begin plotting by organizing x-axis column names into calendar format
- Assign 'mean_calories' to y-axis
- Fill according to 'weekday'

- Assign 'geom_col'
- Add title and caption
- Add X and Y axis labels
- Assign appropriate annotation to each 'weekday'

```
data_df %>%
  group_by(weekday) %>%
  summarize(mean_calories = mean(calories)) %>%
  ggplot(mapping = aes(x = factor(weekday, level =
                                    c('Sunday', 'Monday', 'Tuesday',
                                      'Wednesday', 'Thursday', 'Friday',
                                      'Saturday')),
                       y = mean_calories, fill = weekday)) +
  geom_col() +
  labs(title = "Calories Burned by Weekday",
       caption = "Data Collected in 2016") +
  xlab("Weekday") + ylab("Average Calories Burned") +
  annotate("text", x = "Friday", y = 2430, label = "Friday") +
  annotate("text", x = "Saturday", y = 2618, label = "Saturday") +
  annotate("text", x = "Sunday", y = 2377, label = "Sunday") +
  annotate("text", x = "Monday", y = 2565, label = "Monday") +
  annotate("text", x = "Tuesday", y = 2596, label = "Tuesday") +
  annotate("text", x = "Wednesday", y = 2478, label = "Wednesday") +
  annotate("text", x = "Thursday", y = 2448, label = "Thursday")
```



According to the graph;

- Saturday is when the MOST calories are burned

- Sunday is when the LEAST calories are burned

AVERAGE MINUTES OF SLEEP BY DAY

First, find the MAX of the MEAN for each 'weekday'.

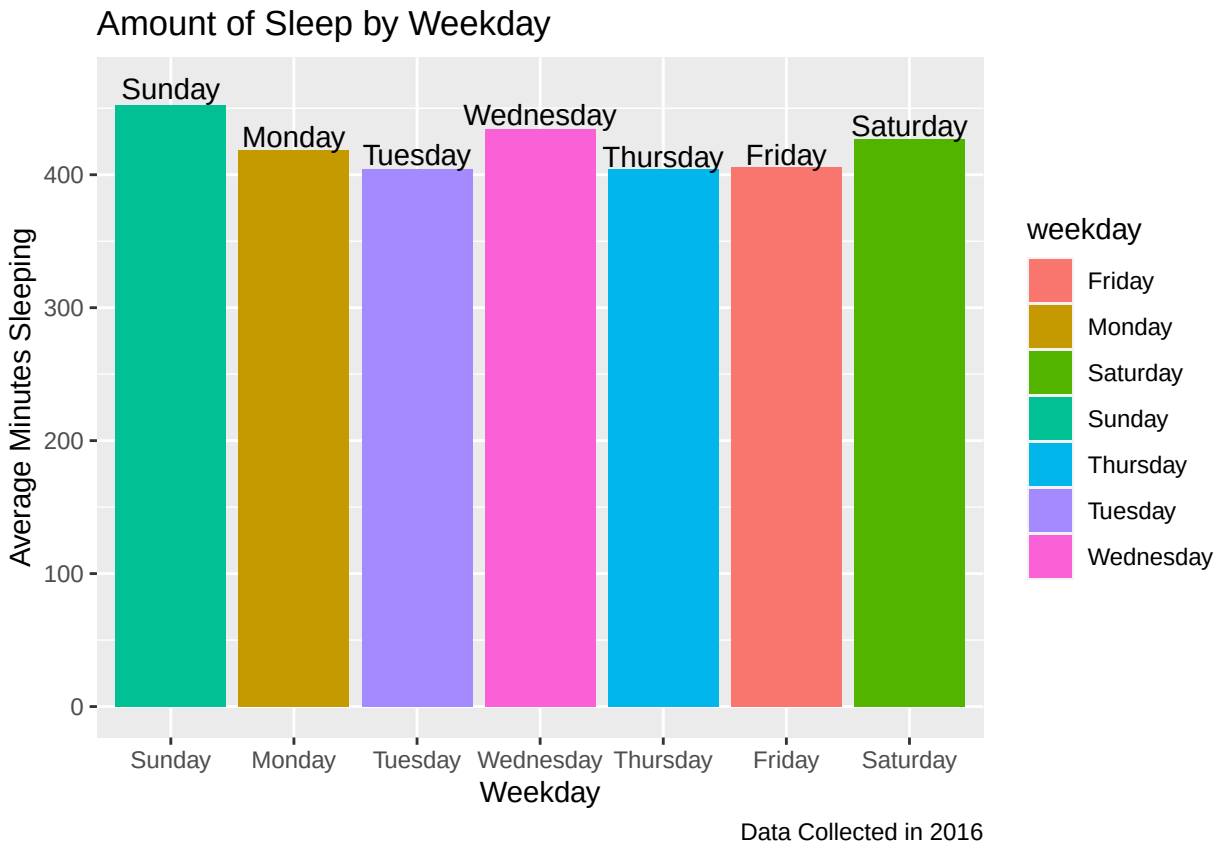
```
data_df %>%  group_by(weekday) %>% summarize(max(mean(totalminutesasleep)))
```

```
## # A tibble: 7 x 2
##   weekday   `max(mean(totalminutesasleep))`
##   <chr>                               <dbl>
## 1 Friday                               405.
## 2 Monday                               419.
## 3 Saturday                             427.
## 4 Sunday                               453.
## 5 Thursday                             404.
## 6 Tuesday                              405.
## 7 Wednesday                            435.
```

Next,

- Group by 'weekday'
- Summarize the MEAN of 'totalminutesasleep' into new variable 'mean_sleep'
- Begin plotting by organizing x-axis column names into calendar format
- Assign 'mean_sleep' to y-axis
- Fill according to 'weekday'
- Assign 'geom_col'
- Add title and caption
- Add X and Y axis labels
- Assign appropriate annotation to each 'weekday'

```
data_df %>%
  group_by(weekday) %>%
  summarize(mean_sleep = mean(totalminutesasleep)) %>%
  ggplot(mapping = aes(x = factor(weekday, level =
                                    c('Sunday', 'Monday', 'Tuesday',
                                      'Wednesday', 'Thursday', 'Friday',
                                      'Saturday')),
                       y = mean_sleep, fill = weekday)) +
  geom_col() +
  labs(title = "Amount of Sleep by Weekday",
       caption = "Data Collected in 2016") +
  xlab("Weekday") + ylab("Average Minutes Sleeping") +
  annotate("text", x = "Friday", y = 415, label = "Friday") +
  annotate("text", x = "Saturday", y = 437, label = "Saturday") +
  annotate("text", x = "Sunday", y = 465, label = "Sunday") +
  annotate("text", x = "Monday", y = 429, label = "Monday") +
  annotate("text", x = "Tuesday", y = 415, label = "Tuesday") +
  annotate("text", x = "Wednesday", y = 445, label = "Wednesday") +
  annotate("text", x = "Thursday", y = 414, label = "Thursday")
```



According to the graph;

- Sunday is when users get the MOST sleep
- Thursday is when users get the LEAST sleep

AVERAGE MINUTES TO FALL ASLEEP BY DAY

First, find the MAX of the MEAN for each 'weekday'.

```
data_df %>%  group_by(weekday) %>% summarize(max(mean(timeawakeinbed)))
```

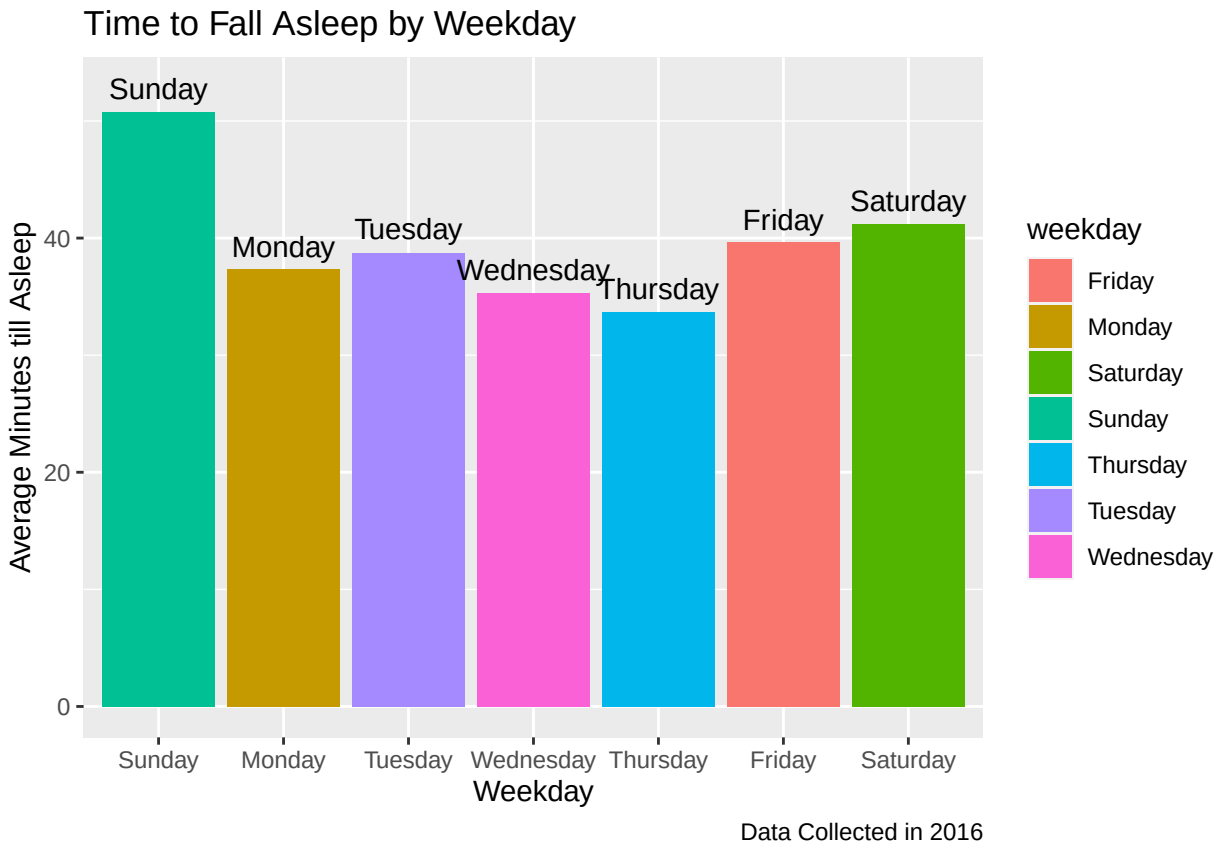
```
## # A tibble: 7 x 2
##   weekday   `max(mean(timeawakeinbed))`
##   <chr>                           <dbl>
## 1 Friday                           39.6
## 2 Monday                           37.3
## 3 Saturday                         41.2
## 4 Sunday                           50.8
## 5 Thursday                         33.7
## 6 Tuesday                          38.8
## 7 Wednesday                        35.3
```

Next,

- Group by 'weekday'
- Summarize the MEAN of 'timeawakeinbed' into new variable 'mean_time_to_sleep'
- Begin plotting by organizing x-axis column names into calendar format
- Assign 'mean_time_to_sleep' to y-axis
- Fill according to 'weekday'

- Assign 'geom_col'
- Add title and caption
- Add X and Y axis labels
- Assign appropriate annotation to each 'weekday'

```
data_df %>%
  group_by(weekday) %>%
  summarize(mean_time_to_sleep = mean(timeawakeinbed)) %>%
  ggplot(mapping = aes(x = factor(weekday, level =
                                    c('Sunday', 'Monday', 'Tuesday',
                                      'Wednesday', 'Thursday', 'Friday',
                                      'Saturday')),
                       y = mean_time_to_sleep, fill = weekday)) +
  geom_col() +
  labs(title = "Time to Fall Asleep by Weekday",
       caption = "Data Collected in 2016") +
  xlab("Weekday") + ylab("Average Minutes till Asleep") +
  annotate("text", x = "Friday", y = 41.6, label = "Friday") +
  annotate("text", x = "Saturday", y = 43.2, label = "Saturday") +
  annotate("text", x = "Sunday", y = 52.8, label = "Sunday") +
  annotate("text", x = "Monday", y = 39.3, label = "Monday") +
  annotate("text", x = "Tuesday", y = 40.8, label = "Tuesday") +
  annotate("text", x = "Wednesday", y = 37.3, label = "Wednesday") +
  annotate("text", x = "Thursday", y = 35.7, label = "Thursday")
```



According to the graph;

- Thursday is when users fall asleep QUICKEST

- Sunday is when users fall asleep SLOWEST

AVERAGE MINUTES DEVICE IS WORN BY DAY

First, find the MAX of the MEAN for each 'weekday'.

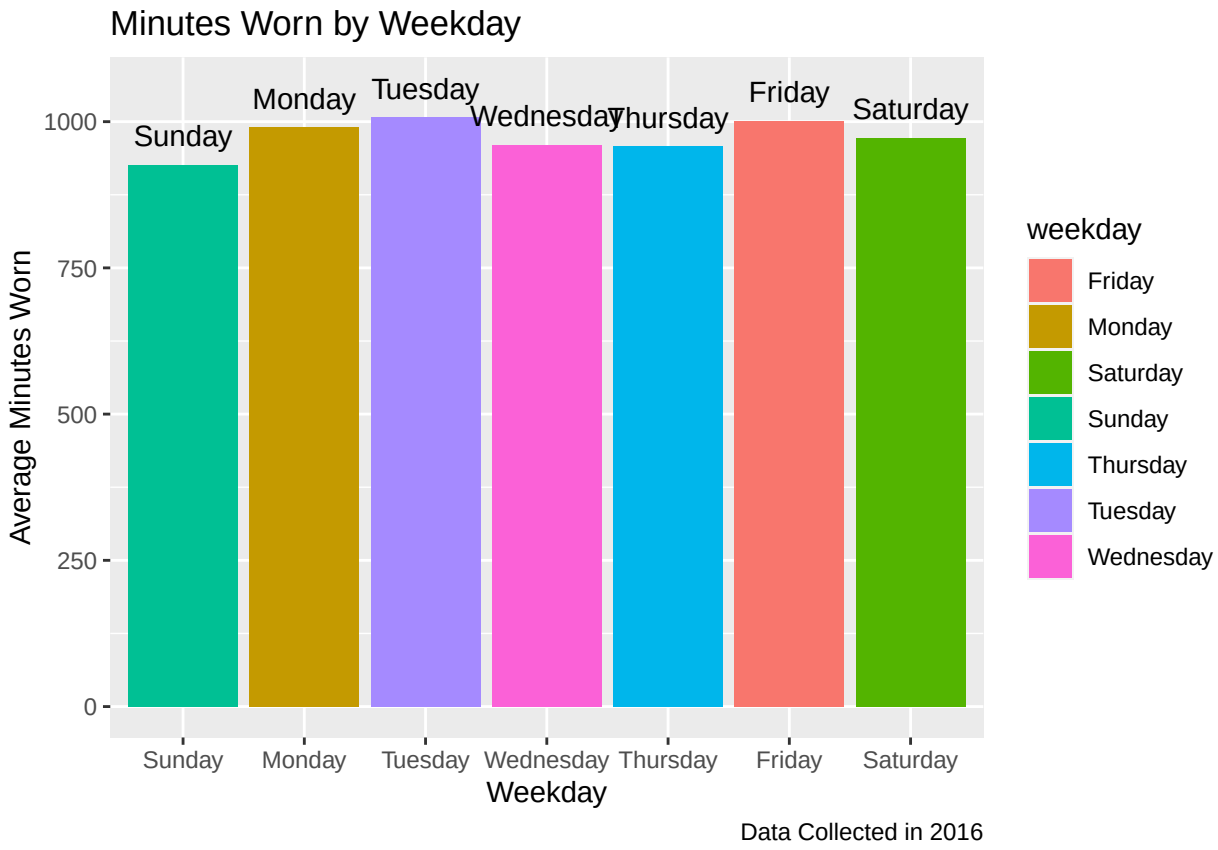
```
data_df %>%  group_by(weekday) %>% summarize(max(mean(minutesworn)))
```

```
## # A tibble: 7 x 2
##   weekday   `max(mean(minutesworn))`
##   <chr>                        <dbl>
## 1 Friday                       1002.
## 2 Monday                        990.
## 3 Saturday                      972.
## 4 Sunday                        927.
## 5 Thursday                      958.
## 6 Tuesday                      1007.
## 7 Wednesday                     960.
```

Next,

- Group by 'weekday'
- Summarize the MEAN of 'minutesworn' into new variable 'mean_wear'
- Begin plotting by organizing x-axis column names into calendar format
- Assign 'mean_wear' to y-axis
- Fill according to 'weekday'
- Assign 'geom_col'
- Add title and caption
- Add X and Y axis labels
- Assign appropriate annotation to each 'weekday'

```
data_df %>%
  group_by(weekday) %>%
  summarize(mean_wear = mean(minutesworn)) %>%
  ggplot(mapping = aes(x = factor(weekday, level =
                                    c('Sunday', 'Monday', 'Tuesday',
                                      'Wednesday', 'Thursday', 'Friday',
                                      'Saturday')),
                       y = mean_wear, fill = weekday)) +
  geom_col() +
  labs(title = "Minutes Worn by Weekday",
       caption = "Data Collected in 2016") +
  xlab("Weekday") + ylab("Average Minutes Worn") +
  annotate("text", x = "Friday", y = 1052, label = "Friday") +
  annotate("text", x = "Saturday", y = 1022, label = "Saturday") +
  annotate("text", x = "Sunday", y = 977, label = "Sunday") +
  annotate("text", x = "Monday", y = 1040, label = "Monday") +
  annotate("text", x = "Tuesday", y = 1057, label = "Tuesday") +
  annotate("text", x = "Wednesday", y = 1010, label = "Wednesday") +
  annotate("text", x = "Thursday", y = 1008, label = "Thursday")
```



According to the graph;

- Tuesday is when users wear the device the MOST
- Sunday is when users wear the device to LEAST
- We find consistant usage numbers throughout the week

PLOTTING BY 'alldaywear'

AVERAGE ACTIVE MINUTES BY WEAR VS NON-WEAR

First, find the MAX of the MEAN for 'TRUE' and 'FALSE' in 'alldaywear'.

```
data_df %>%  group_by(alldaywear) %>% summarize(max(mean(activeminutes)))
```

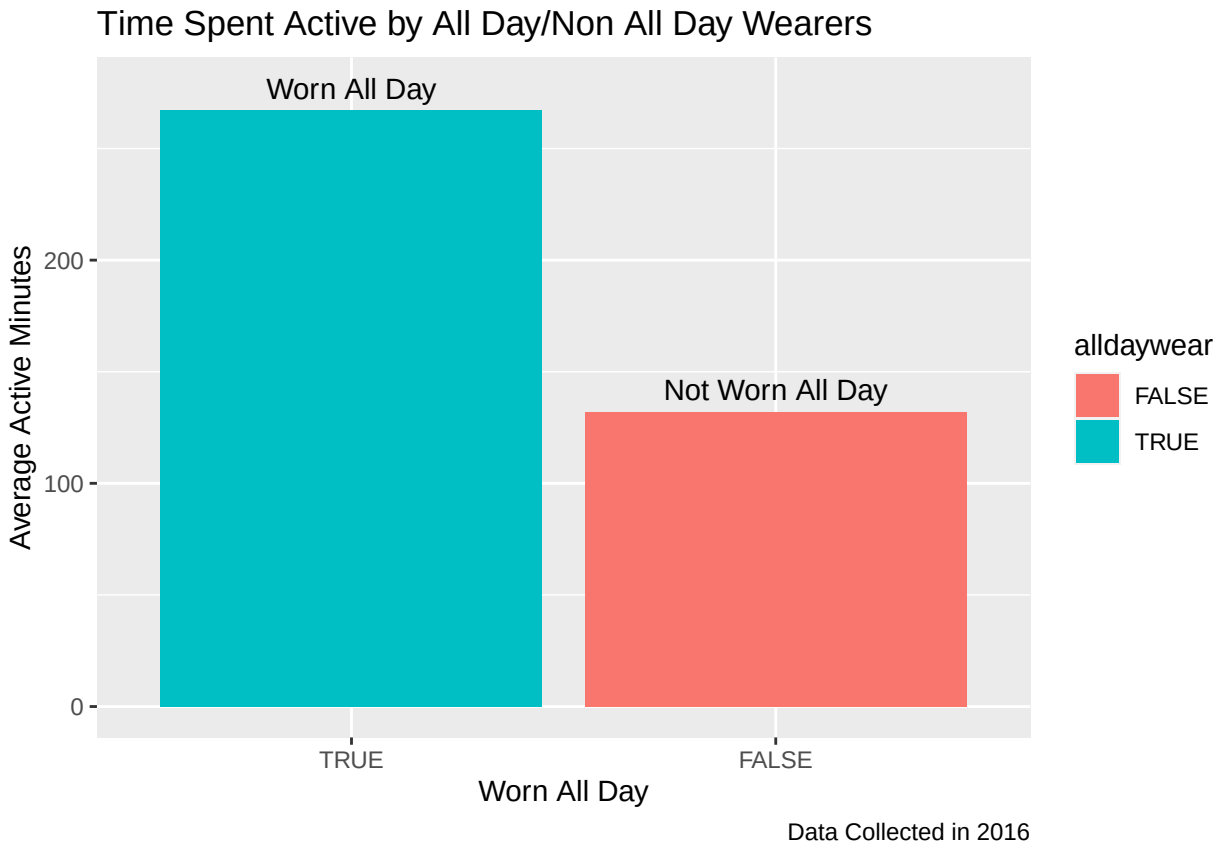
```
## # A tibble: 2 x 2
##   alldaywear `max(mean(activeminutes))`
##   <lgl>                           <dbl>
## 1 FALSE                            132
## 2 TRUE                             267.
```

Next,

- Group by 'alldaywear'
- Summarize the MEAN of 'activeminutes' into new variable 'mean_active'
- Begin plotting by organizing x-axis column names into calendar format
- Assign 'mean_active' to y-axis
- Fill according to 'alldaywear'
- Assign 'geom_col'
- Add title and caption

- Add X and Y axis labels
- Assign appropriate annotation to each column

```
data_df %>%
  group_by(alldaywear) %>%
  summarize(mean_active = mean(activeminutes)) %>%
  ggplot(mapping = aes(x = factor(alldaywear, level =
                                    c('TRUE', 'FALSE')),
                       y = mean_active, fill = alldaywear)) +
  geom_col() +
  labs(title = "Time Spent Active by All Day/Non All Day Wearers",
       caption = "Data Collected in 2016") +
  xlab("Worn All Day") + ylab("Average Active Minutes") +
  annotate("text", x = "TRUE", y = 277, label = "Worn All Day") +
  annotate("text", x = "FALSE", y = 142, label = "Not Worn All Day")
```



According to the graph;

- All Day Wearers have a HIGHER average active minutes
- There could just be more data recorded, increasing the average.

AVERAGE CALORIES BURNED BY WEAR VS NON-WEAR

First, find the MAX of the MEAN for 'TRUE' and 'FALSE' in 'alldaywear'.

```
data_df %>%  group_by(alldaywear) %>% summarize(max(mean(calories)))
```

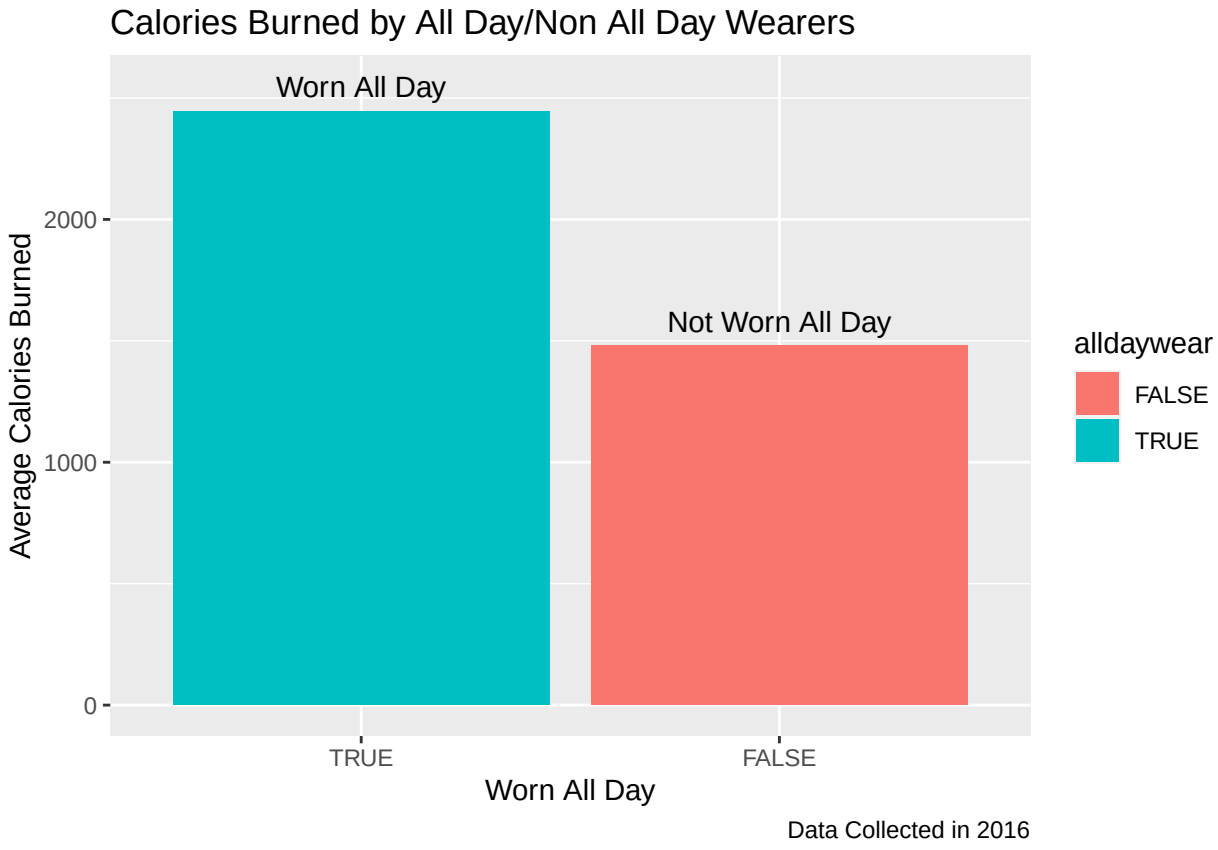
```
## # A tibble: 2 x 2
##   alldaywear `max(mean(calories))`
```

```
##   <lgl>                     <dbl>
## 1 FALSE                     1480.
## 2 TRUE                      2446.
```

Next,

- Group by 'alldaywear'
- Summarize the MEAN of 'calories' into new variable 'mean_calories'
- Begin plotting by organizing x-axis column names into calendar format
- Assign 'mean_calories' to y-axis
- Fill according to 'alldaywear'
- Assign 'geom_col'
- Add title and caption
- Add X and Y axis labels
- Assign appropriate annotation to each column

```
data_df %>%
  group_by(alldaywear) %>%
  summarize(mean_calories = mean(calories)) %>%
  ggplot(mapping = aes(x = factor(alldaywear, level =
                                    c('TRUE', 'FALSE')),
                       y = mean_calories, fill = alldaywear)) +
  geom_col() +
  labs(title = "Calories Burned by All Day/Non All Day Wearers",
       caption = "Data Collected in 2016") +
  xlab("Worn All Day") + ylab("Average Calories Burned") +
  annotate("text", x = "TRUE", y = 2546, label = "Worn All Day") +
  annotate("text", x = "FALSE", y = 1580, label = "Not Worn All Day")
```



According to the graph;

- All Day Wearers averaged burning MORE calories
- There could be more data recorded, increasing the average

AVERAGE MINUTES OF SLEEP BY WEAR VS NON-WEAR

First, find the MAX of the MEAN for 'TRUE' and 'FALSE' in 'alldaywear'.

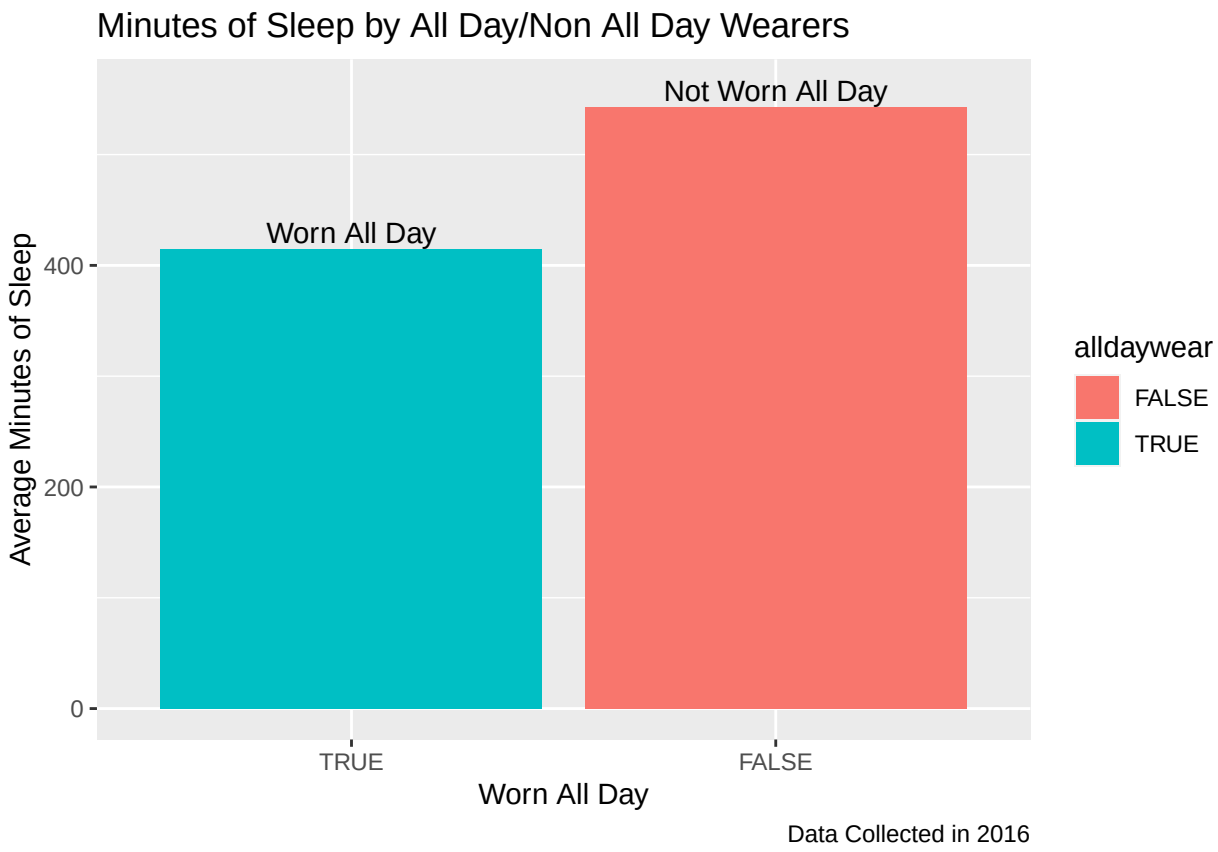
```
data_df %>%  group_by(alldaywear) %>% summarize(max(mean(totalminutesasleep)))
```

```
## # A tibble: 2 x 2
##   alldaywear `max(mean(totalminutesasleep))`
##   <lgl>                                <dbl>
## 1 FALSE                                 543.
## 2 TRUE                                  415.
```

Next,

- Group by 'alldaywear'
- Summarize the MEAN of 'totalminutesasleep' into new variable 'mean_sleep'
- Begin plotting by organizing x-axis column names into calendar format
- Assign 'mean_sleep' to y-axis
- Fill according to 'alldaywear'
- Assign 'geom_col'
- Add title and caption
- Add X and Y axis labels
- Assign appropriate annotation to each column

```
data_df %>%
  group_by(alldaywear) %>%
  summarize(mean_sleep = mean(totalminutesasleep)) %>%
  ggplot(mapping = aes(x = factor(alldaywear, level =
                                    c('TRUE', 'FALSE')),
                       y = mean_sleep, fill = alldaywear)) +
  geom_col() +
  labs(title = "Minutes of Sleep by All Day/Non All Day Wearers",
       caption = "Data Collected in 2016") +
  xlab("Worn All Day") + ylab("Average Minutes of Sleep") +
  annotate("text", x = "TRUE", y = 430, label = "Worn All Day") +
  annotate("text", x = "FALSE", y = 558, label = "Not Worn All Day")
```



According to the graph;

- All Day Wearers get LESS sleep

AVERAGE MINUTES TO FALL ASLEEP BY WEAR VS NON-WEAR

First, find the MAX of the MEAN for 'TRUE' and 'FALSE' in 'alldaywear'.

```
data_df %>%  group_by(alldaywear) %>% summarize(max(mean(timeawakeinbed)))
```

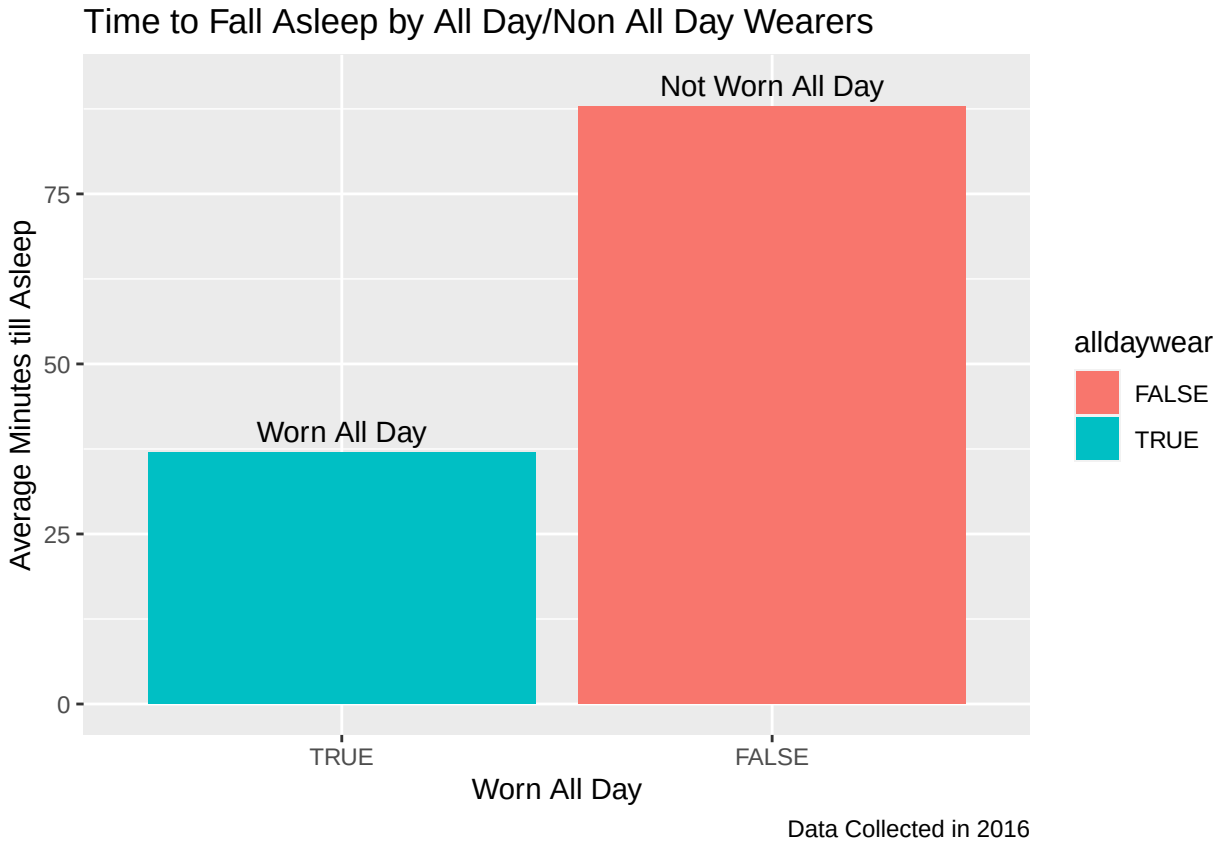
```
## # A tibble: 2 x 2
##   alldaywear `max(mean(timeawakeinbed))`
##   <lgl>                            <dbl>
## 1 FALSE                             87.9
## 2 TRUE                              37.0
```

Next,

- Group by 'alldaywear'
- Summarize the MEAN of 'timeawakeinbed' into new variable 'mean_time_to_sleep'
- Begin plotting by organizing x-axis column names into calendar format
- Assign 'mean_time_to_sleep' to y-axis
- Fill according to 'alldaywear'
- Assign 'geom_col'
- Add title and caption
- Add X and Y axis labels
- Assign appropriate annotation to each column

```
data_df %>%
  group_by(alldaywear) %>%
```

```
summarize(mean_time_to_sleep = mean(timeawakeinbed)) %>%
ggplot(mapping = aes(x = factor(alldaywear, level =
                                  c('TRUE', 'FALSE')),
                     y = mean_time_to_sleep, fill = alldaywear)) +
geom_col() +
labs(title = "Time to Fall Asleep by All Day/Non All Day Wearers",
     caption = "Data Collected in 2016") +
xlab("Worn All Day") + ylab("Average Minutes till Asleep") +
annotate("text", x = "TRUE", y = 40, label = "Worn All Day") +
annotate("text", x = "FALSE", y = 90.9, label = "Not Worn All Day")
```



According to the graph;

- All Day Wearers are QUICKER to fall asleep

SUMMARY

There were certainly some trends to be found in the data being analyzed weekly.

Daily activity tracking showed that both measured activity and calories burned are at their highest on Saturday and lowest on Sunday.

Daily sleep tracking showed that Sunday is when user get the most sleep and Thursday they get the least. Interesting enough, this is reversed when considering the time to fall asleep. On Thursday, the day with the least sleep, users are quickest to fall asleep, and on Sunday, the day with the most sleep, users fall asleep the slowest.

Regardless of the day, users tended to wear the device on a consistant basis.

When analyzing the difference between All Day Wearers and Non All Day Wearers, we were able to spot some trends as well.

Data showed that across the board, users who wore the device all day, twelve hours or more, recorded more activity and calories burned. I should note that this is more than likely due to having more data being tracked throughout the day. Additional research could be done by breaking down the calories burned per high intensity minute to see which user group is burning more calories in a given recorded session.

When analyzing sleep patterns by user group we find that All Day Wearers tended to get 23% less sleep, but were able to fall asleep an astounding 58% faster.

CASE USE

- Advertise leisure products to users on days they are least active; top 3 being Sunday, Wednesday, and Thursday
- Advertise exercise products to users on days they are most active; top 3 being Monday, Tuesday, and Saturday
- Advertise sleep aid products to users on days where they got the least sleep; top 3 being Tuesday, Thursday, and Friday
- Advertise exercise products geared towards someone who wears the device all day; i.e. new faces and bands
- Advertise sleep aid products geared towards someone who wears the device all day; i.e. quicker or longer sleep aid

CONSIDERATIONS

- Further insight could be gathered via survey to device users to record gender, age, and sexuality to more accurately understand the variations in demographic
- More detailed data analysis could be performed that targets calories burned by activity intensity level to determine which user group burns the most calories while device is worn for recorded exercise
- A larger grouping of user data for weight loss/gain could allow for further analysis as to which groups of users are seeing the most progress in their exercise, and could be further analyzed for trends related to the currently appearing trend of “More Activity, More Sleep”