Fitness_tracker_project1

Ryan King

12/4/2021

Project 1: Fitness tracker

Load the data and environment

```
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)
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
library(stringr)
file_path <- "C:/Users/I0485672/Downloads/activity.csv"
data <- read.csv(file_path)
```

Analyzing the data

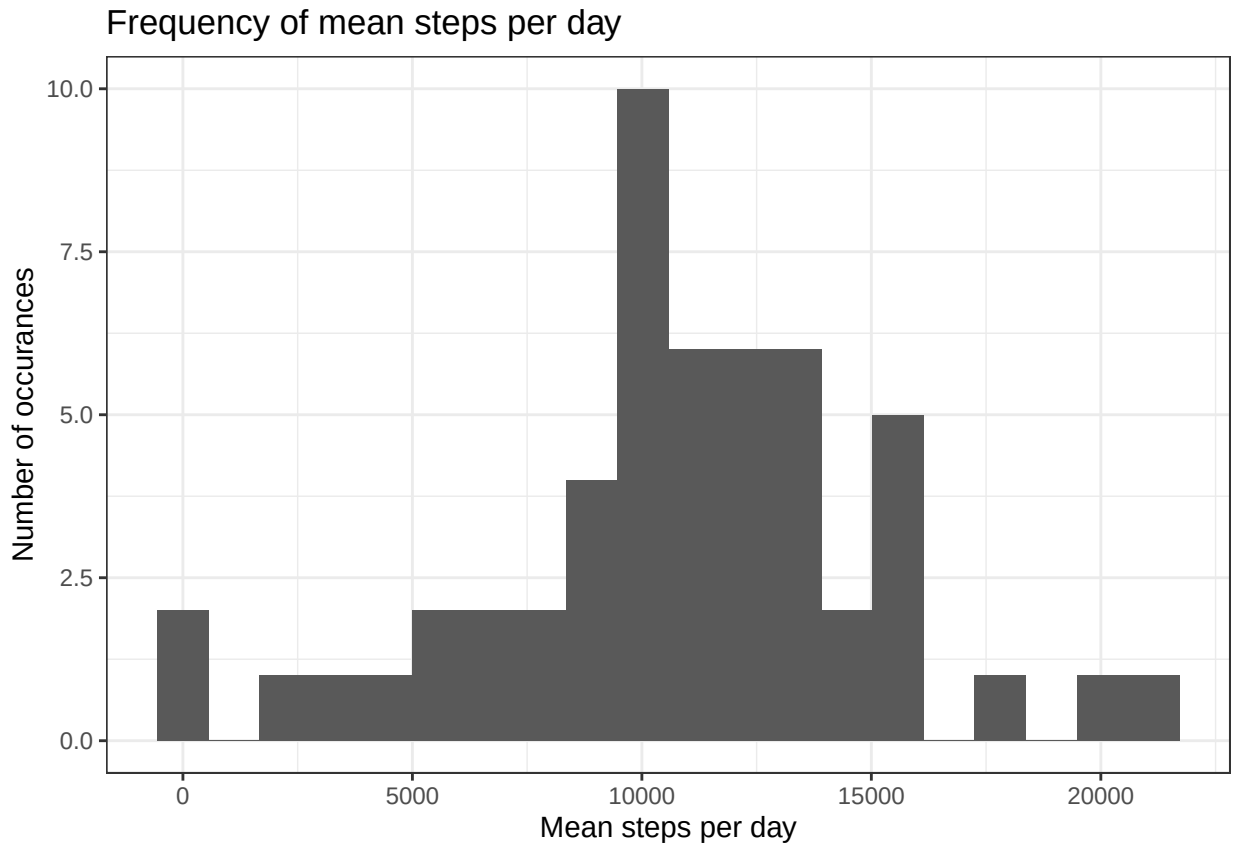
There are some NA's in here. They will be removed until imputed step. Why? They don't have the device on or working. We cannot say that means zero steps.

Let's look at the total number of steps taken each day

```
data_summary <- data[complete.cases(data), ] %>% group_by(date) %>% summarize(mean=sum(steps, na.rm = TR
data_summary
```

```
## # A tibble: 53 x 2
##    date        mean
##    <chr>      <int>
##  1 2012-10-02   126
##  2 2012-10-03 11352
##  3 2012-10-04 12116
##  4 2012-10-05 13294
##  5 2012-10-06 15420
##  6 2012-10-07 11015
##  7 2012-10-09 12811
##  8 2012-10-10  9900
##  9 2012-10-11 10304
## 10 2012-10-12 17382
## # ... with 43 more rows
```

```
data_summary %>% ggplot(aes(x = mean)) + geom_histogram(bins = 20) +
  theme_bw() + labs(x='Mean steps per day', y='Number of occurances',
                    title = 'Frequency of mean steps per day')
```



let's look at the mean and steps per day.

```
data_summary <-  data[complete.cases(data), ] %>% group_by(date) %>% summarize(StepsSum=sum(steps, na.rm
mean(data_summary$StepsSum)
```

```
## [1] 10766.19
```

```
median(data_summary$StepsSum)
```

```
## [1] 10765
```

What's the max steps in a given interval?

```
data_summary.max <-  data[complete.cases(data), ] %>% group_by(interval) %>% summarize(StepsMax=max(step
tophit <- data_summary.max %>% arrange(desc(StepsMax)) %>% head(n=1)
tophit
```

```
## # A tibble: 1 x 2
##   interval StepsMax
##      <int>    <int>
## 1      615      806
```

The 615 interval has the max steps

But that's max... what's the max (highest) average steps taken

```
data_summary <-  data[complete.cases(data), ] %>% group_by(interval) %>% summarize(mean=mean(steps, na.r
data_summary %>% arrange(desc(mean))
```

```
## # A tibble: 288 x 2
##    interval  mean
##       <int> <dbl>
##  1      835  206.
##  2      840  196.
##  3      850  183.
##  4      845  180.
##  5      830  177.
##  6      820  171.
##  7      855  167.
##  8      815  158.
##  9      825  155.
## 10      900  143.
## # ... with 278 more rows
```

```
max.avg.results <- data_summary %>% arrange(desc(mean)) %>% head(n=1)
print(paste("The largest interval is,", max.avg.results$interval, "with", max.avg.results$mean, "max ste
```

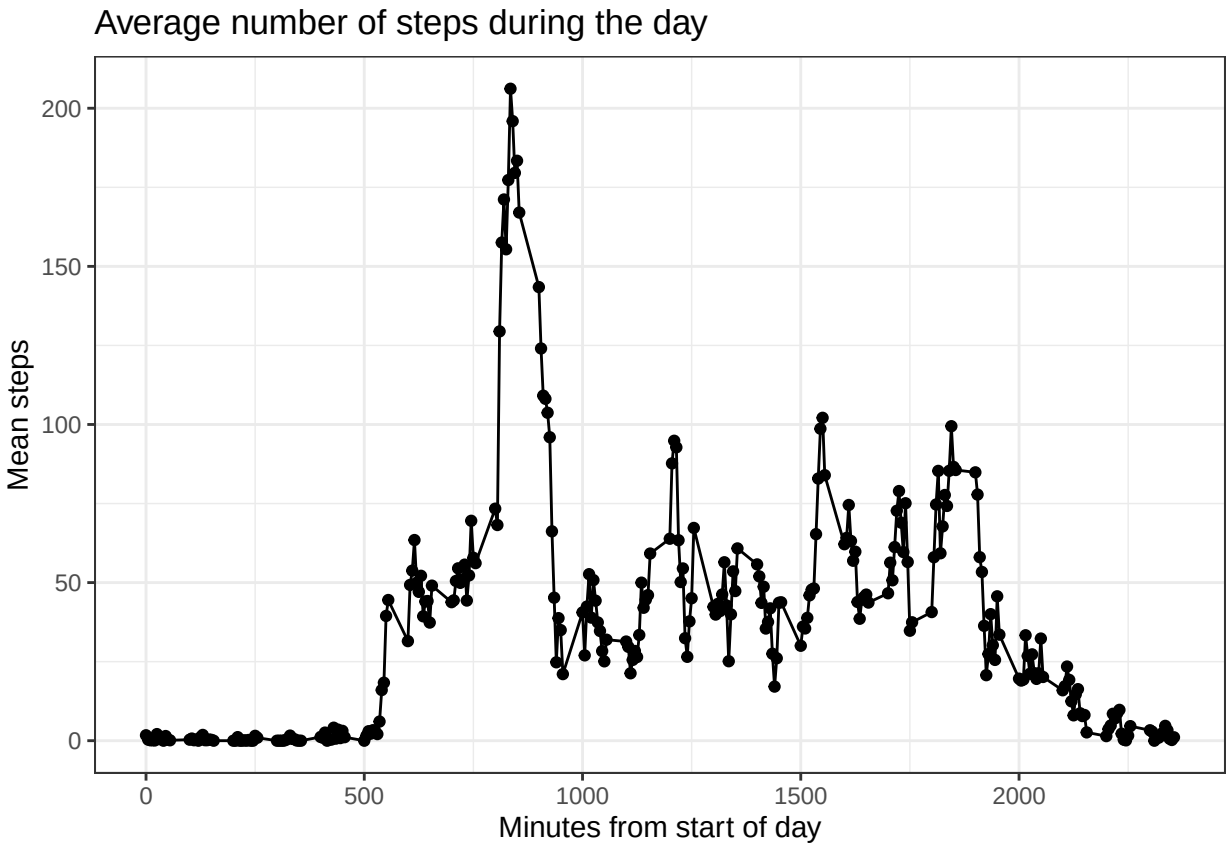
```
## [1] "The largest interval is, 835 with 206.169811320755 max steps on average"
```

The largest interval is, 835 with 206.169811320755 max steps on average

let's look at a time series

```
data_summary <-  data[complete.cases(data), ] %>% group_by(interval) %>% summarize(mean=mean(steps, na.

ggplot(data_summary, aes(x = interval, y = mean)) +
  geom_point() + geom_line() + theme_bw() +
  labs(y="Mean steps", x = "Minutes from start of day",
       title = "Average number of steps during the day")
```



Imputing data

How to impute data? There's many many theories out there 1. mean - on each interval 2. nearest neighbor - for example, steps on a Saturday vs. Monday and take mean 3. So much more, but #2 is really good.

Let's do the mean of the days of the week for the given interval If you really wanted a nice imputation, perhaps a lagging average?

```
data.avg.per.weekday <- data
data.avg.per.weekday$date <- as.Date(data$date) %>% weekdays()
data.avg.per.weekday <-  data.avg.per.weekday[complete.cases(data.avg.per.weekday), ] %>% group_by(date
```

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

```
data.meanImpute <- data
for (i in 1:nrow(data.meanImpute)){
  if(is.na(data.meanImpute[i, 1])){
    data.meanImpute[i, 1] <- data.avg.per.weekday %>%
      filter(date == weekdays(as.Date(data[i, 2])) & interval == data[i, 3]) %>%
      .[[3]]
  }
}

print("Checking data:")
```

```
## [1] "Checking data:"
```

```
any(is.na(data.meanImpute))
```

```
## [1] FALSE
```

```
# cha-ching
```

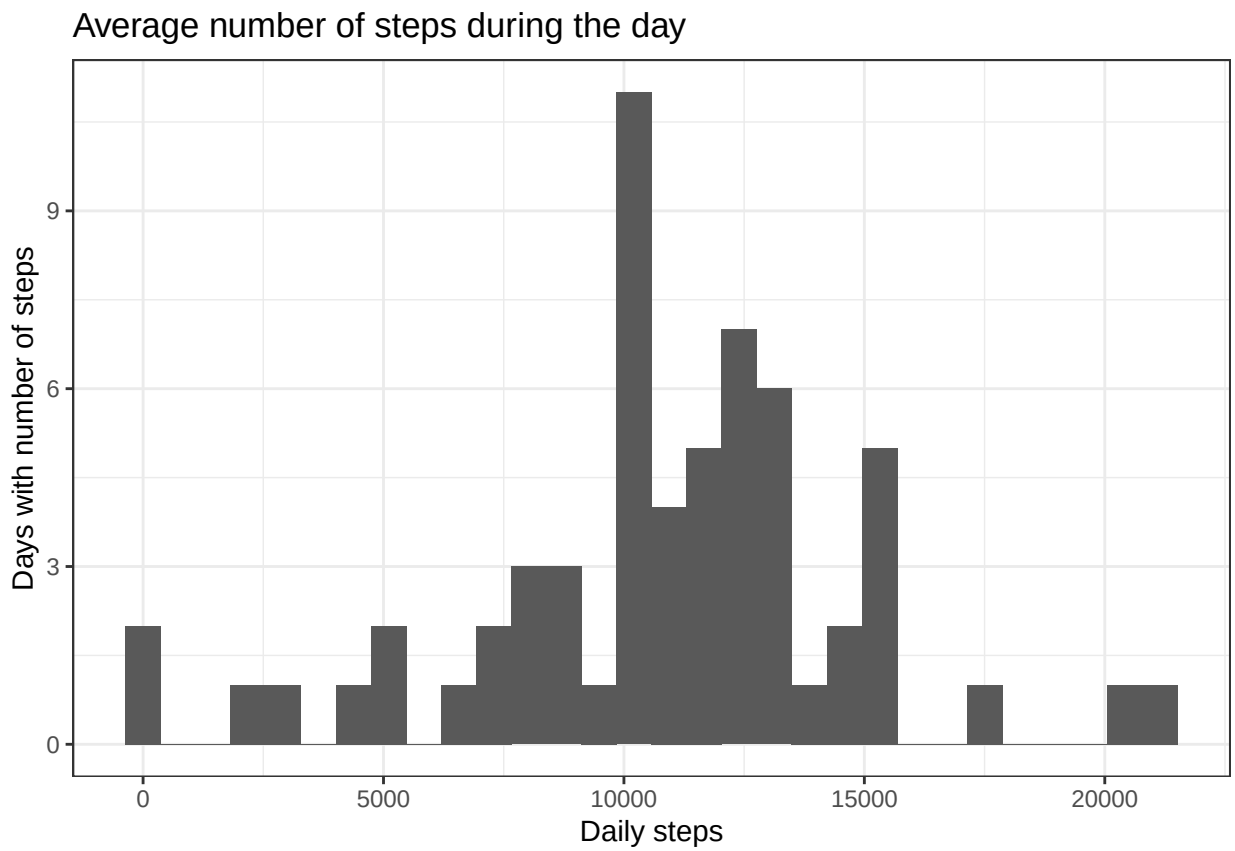
Visualizing imputed data

Let's make a histogram of the total number of steps taken each day after missing values are imputed

```
data.check <- data.meanImpute %>% group_by(date) %>% summarize(DailySteps = sum(steps))

data.check %>% ggplot(aes(x = DailySteps)) +
geom_histogram() + theme_bw() +
  labs(y="Days with number of steps", x = "Daily steps",
       title = "Average number of steps during the day")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



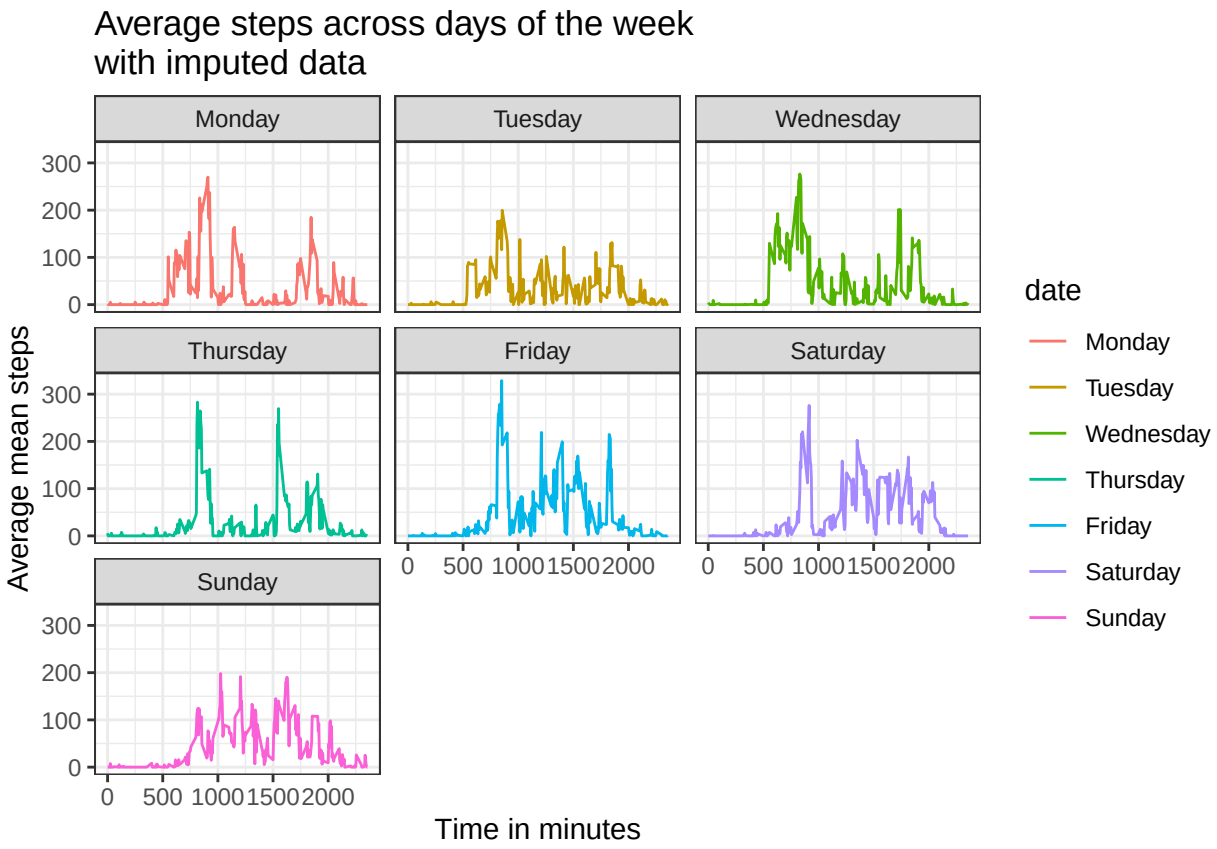
Let's use this for a days of the week plot

```
# let's get the days of the week reordered properly for the graph
data.meanImpute$date <- weekdays(as.Date(data.meanImpute$date))
data.meanImpute$date <- factor(data.meanImpute$date, levels =
                                    c("Monday", "Tuesday", "Wednesday",
                                      "Thursday", "Friday", "Saturday",
                                      "Sunday"))

data.meanImpute.eachday <- data.meanImpute %>% group_by(date, interval) %>% summarise(meanSteps = mean(s
```

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

```
data.meanImpute.eachday %>% ggplot(aes(x = interval, y = meanSteps, color = date)) +
  geom_line() + theme_bw() + facet_wrap(~date) +
  labs(title = "Average steps across days of the week\nwith imputed data",
       x = "Time in minutes", y = "Average mean steps")
```



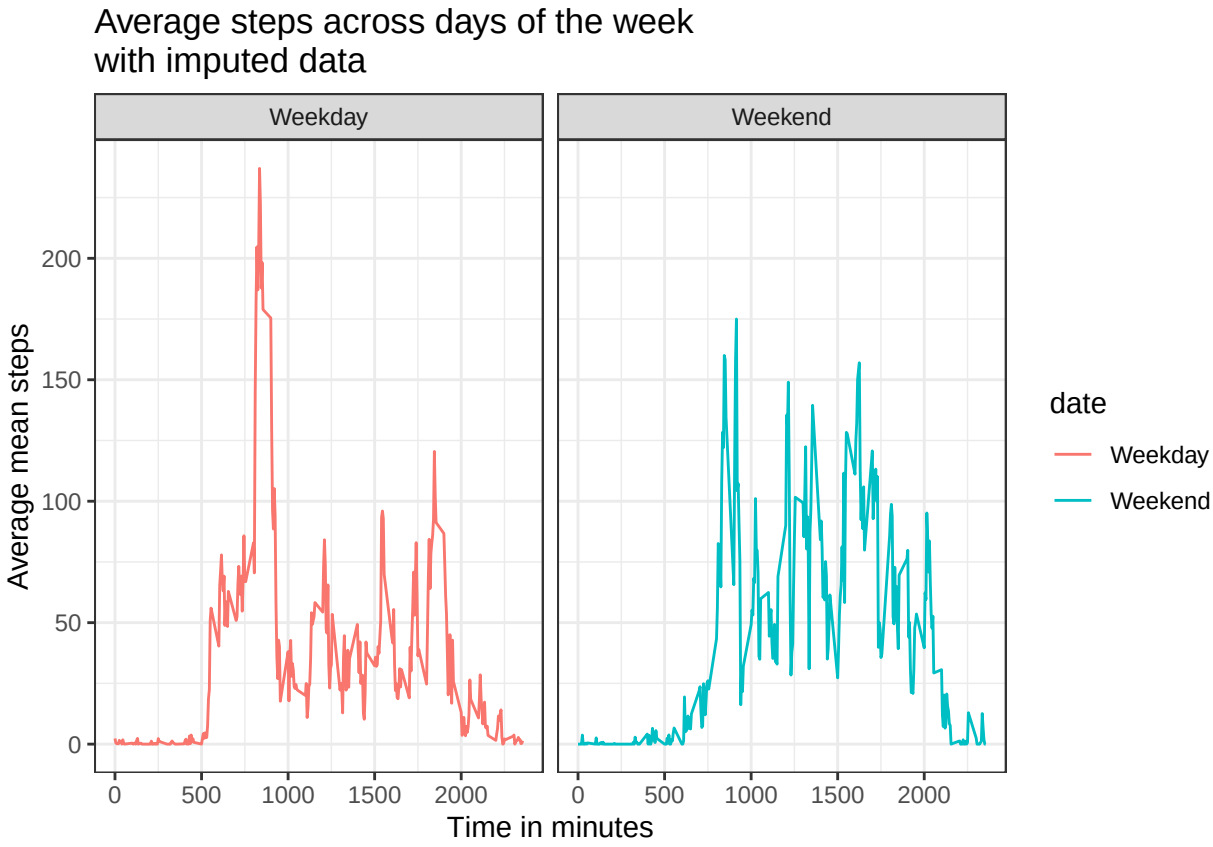
Looks like people are more active saturday night and weekday mornings

They may want weekdays VS weekends, not all days.

```
data.meanImpute$date <- str_replace(data.meanImpute$date, "Saturday", "Weekend")
data.meanImpute$date <- str_replace(data.meanImpute$date, "Sunday", "Weekend")
data.meanImpute$date <- if_else(data.meanImpute$date == 'Weekend', 'Weekend', 'Weekday')
data.meanImpute.final <- data.meanImpute %>% group_by(date, interval) %>% summarise(meanSteps = mean(st
```

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

```
data.meanImpute.final %>% ggplot(aes(x = interval, y = meanSteps, color = date)) +
  geom_line() + theme_bw() + facet_wrap(~date) +
  labs(title = "Average steps across days of the week\nwith imputed data",
       x = "Time in minutes", y = "Average mean steps")
```


Average steps across days of the week
with imputed data
Weekday
Weekend
Average mean steps
200
150
100
50
0
0
500
1000
1500
2000
Time in minutes
date
Weekday
Weekend