Activity_Monitoring_data

James Northup

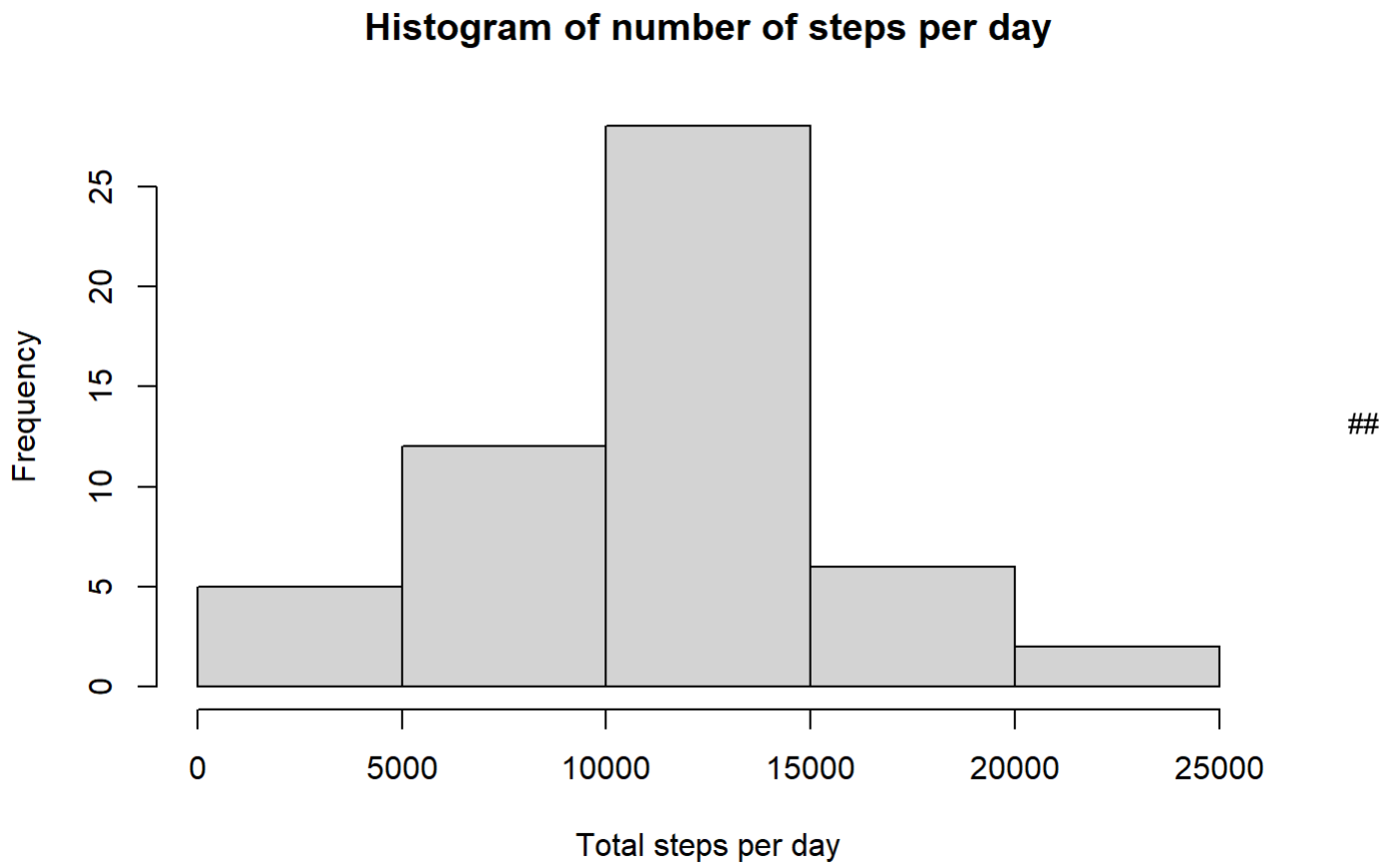
9/23/2020

1.) Code for reading in the dataset and/or processing the data

```
library(tidyverse)

## Create temp file
temp <- tempfile()

## Downloads zipped data
download.file("https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip",
              temp)

## Unzips files
path1 <- unzip(temp, "activity.csv")


## Delete temp file
unlink(temp)

## Reads RDS into data frame
Activity_data <- read.csv(path1)
Activity_data$date <- as.Date(Activity_data$date , format = "%Y-%m-%d")
```

2.) Histogram of the total number of steps taken each day

```
## Histograms with fill based on variable
Steps_per_Day <- aggregate(steps ~ date, Activity_data, sum)


## Create Histogram
hist(Steps_per_Day$steps, xlab = "Total steps per day", main = "Histogram of number of steps per
day")
```



##

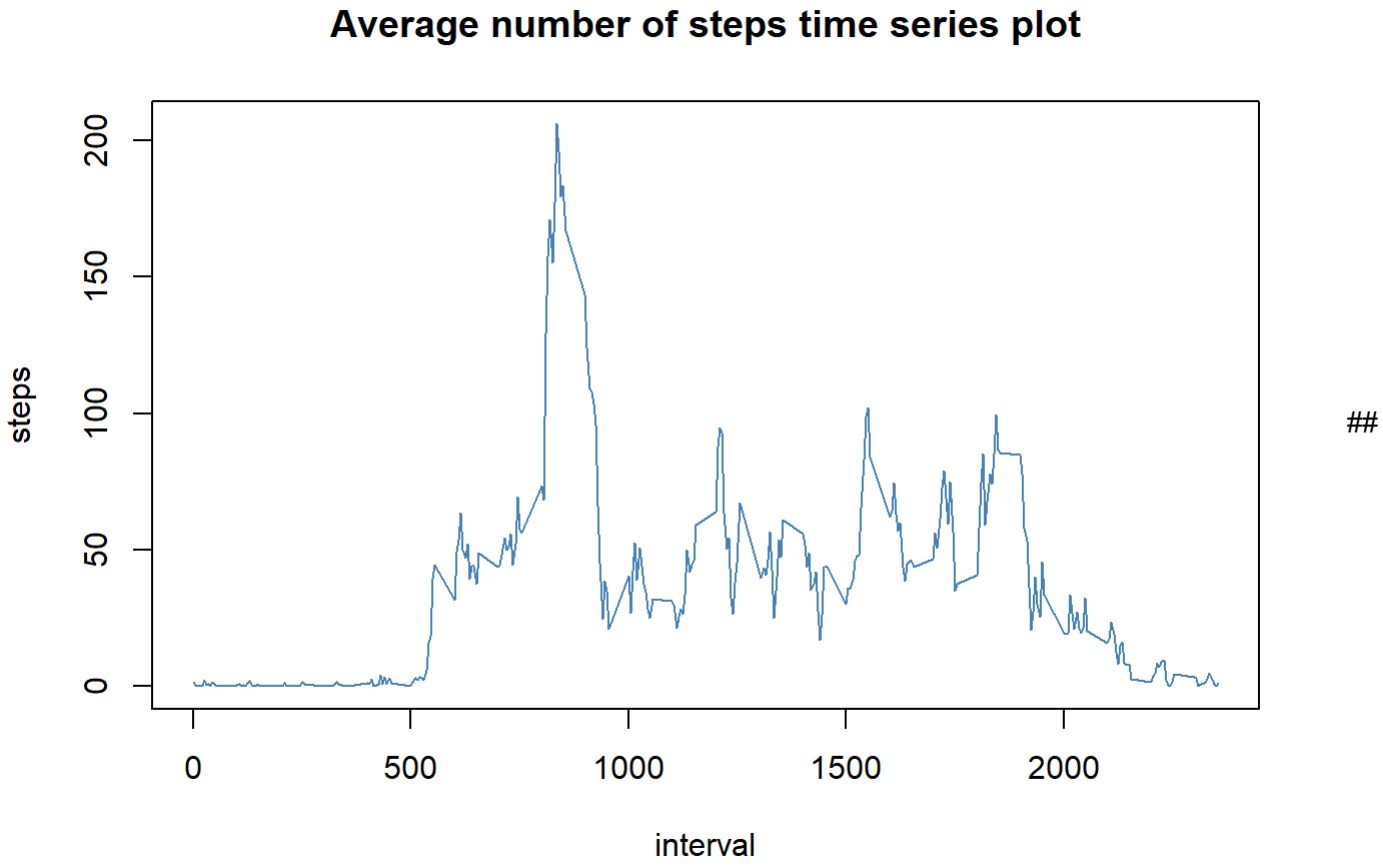
3.) Mean and median number of steps taken each day

```
# Mean steps per day
steps_Day_Mean <- mean(Steps_per_Day$steps)
steps_Day_Mean <- median(Steps_per_Day$steps)
```

The mean for steps per day is 10765 while the median is 10765

4.) Time series plot of the average number of steps taken

```
mean_Steps_per_Interval <-
  aggregate(steps ~ interval, Activity_data, mean)
plot(
  steps ~ interval,
  mean_Steps_per_Interval,
  type = "l",
  main = "Average number of steps time series plot",
  col = "steelblue"
)
```



##

5.) The 5-minute interval that, on average, contains the maximum number of steps

```
## max step in 5-minute interval
max_step_interval <-
  mean_Steps_per_Interval[mean_Steps_per_Interval$steps == max(mean_Steps_per_Interval$steps), ]
```

Maximum steps per 5 minute interval is 206.1698113 at interval 835

6.) Code to describe and show a strategy for imputing missing data

```
## Check if data has missing data
count(Activity_data[is.na(Activity_data$steps), ])
```

```
##      n
## 1 2304
```

```
count(Activity_data[is.na(Activity_data$interval), ])
```

```
##   n
## 1 0
```

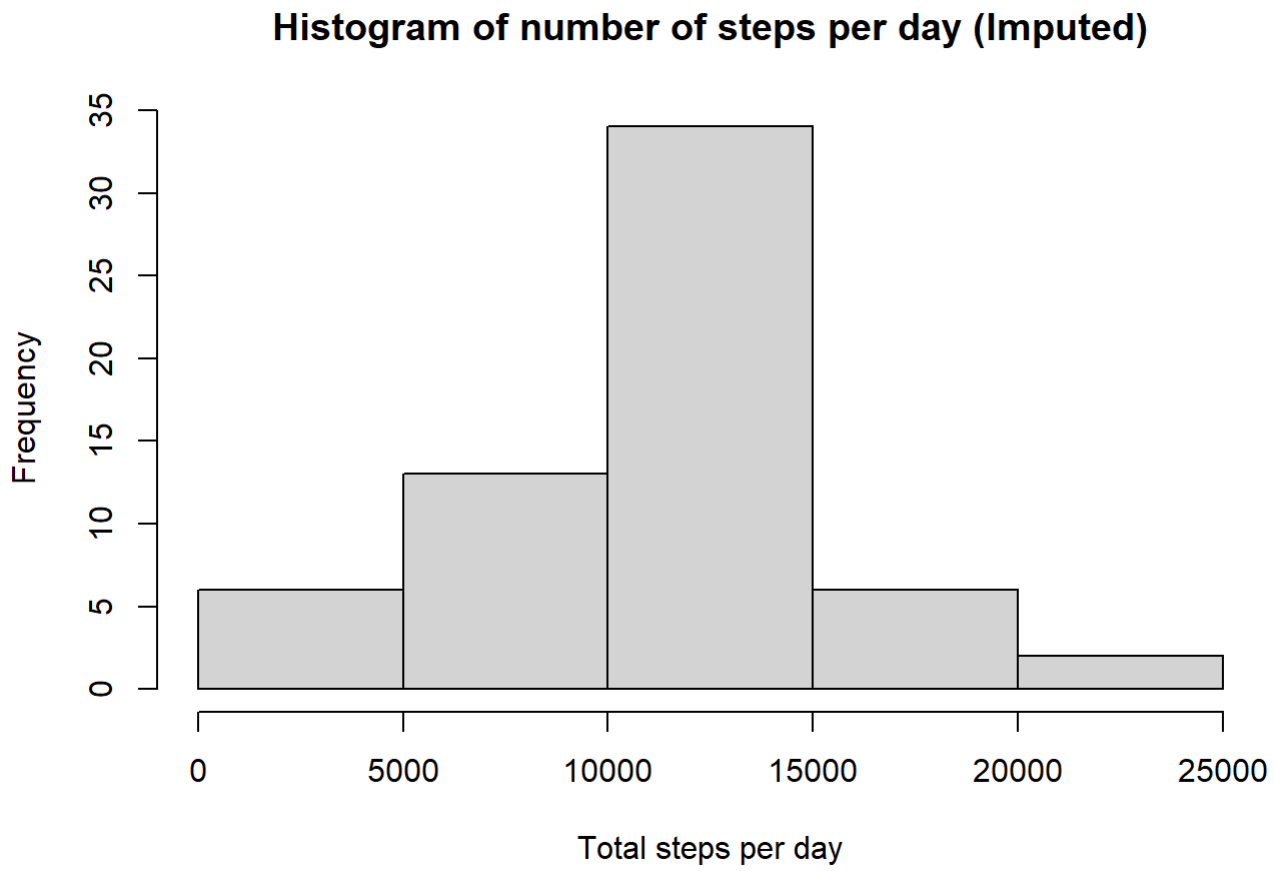
```
count(Activity_data[is.na(Activity_data$date), ])
```

```
##   n
## 1 0
```

```
## NA are present in column steps. Below will replace step na
## values into mean values

## Copy original data into new new variable
Activity_data_NAisMean <- Activity_data

index <- 1

## For loop replaces na steps into mean, if mean is na then we set mean to 0.

for (each in Activity_data_NAisMean$steps) {
  if (is.na(Activity_data_NAisMean$steps[index])) {
    mean_index <-
      mean(Activity_data_NAisMean$steps[1:index], na.rm = TRUE)
    if (is.na(mean_index)) {
      mean_index <- 0
    }
    Activity_data_NAisMean$steps[index] <- mean_index


  }
  index = index + 1

}
```

7.) Histogram of the total number of steps taken each day after missing values are imputed

```
## Histograms with fill based on variable
Steps_per_Day_NAisMean <-
  aggregate(steps ~ date, Activity_data_NAisMean, sum)

## Create Histogram
hist(Steps_per_Day_NAisMean$steps,
     xlab = "Total steps per day",
     main = "Histogram of number of steps per day (Imputed)")
```

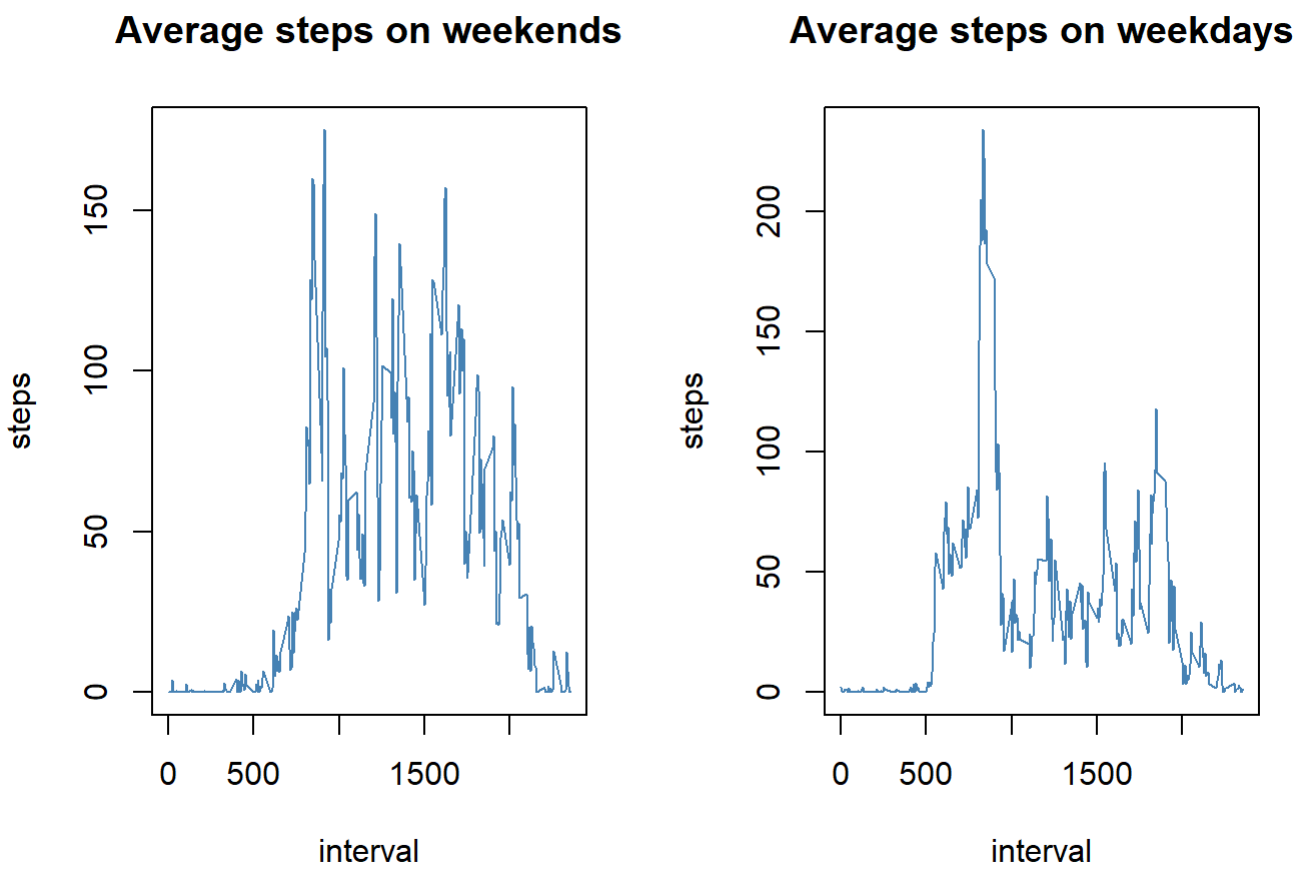


8.) Panel plot comparing the average number of steps taken per 5-minute interval across weekdays and weekends

```
library(chron)
names(Activity_data)
```

```
## [1] "steps"    "date"     "interval"
```

```
mean_Steps_per_Interva_Weekends <-
  aggregate(steps ~ interval, Activity_data[is.weekend(Activity_data$date),], mean)

mean_Steps_per_Interva_Weekdays <-
  aggregate(steps ~ interval, Activity_data[!is.weekend(Activity_data$date),], mean)


par(mfrow=c(1,2))
plot(
  steps ~ interval,
  mean_Steps_per_Interva_Weekends,
  type = "l",
  main = "Average steps on weekends",
  col = "steelblue"
)

plot(
  steps ~ interval,
  mean_Steps_per_Interva_Weekdays,
  type = "l",
  main = "Average steps on weekdays",
  col = "steelblue"
)
```



9.) All of the R code needed to reproduce the results (numbers, plots, etc.) in the report