Peer-graded Assignment - First Course Project

Jonah Winninghoff

7/22/2019

This paper develops based on Peer-graded Assignment: Course Project 1 that will evaluate my ability to manipulate dataset and create a set of plots and histograms. Not only does it evaluates on my ability, it evaluates my ability to use *Markdown* so that it intends to cover:

1.) Coding for reading in the dataset

2.) Histogram of the total of steps per day

3.) Mean and median number of steps per day

4.) Plotting timeline of the average number of steps

5.) 5-minute interval containing the maximum number of steps on average

6.) Code to describe and show how to impute missing data

7.) Post-missing removal histogram of the total number of steps per day

8.) Panel plot in comparsion to weekend and weekday average number of steps per 5-minute interval

From now on, the first part is to demonstrate how to code for reading in the dataset. To identify my getwd() and organize my file both are needed, of course. When this pre-reading process is done, the coding for reading this file is not only a thing needed to do but ggplot2, dplyr are applied. Also, the *date* in dataset is coerced into date and dataset is organized for convenient reading. Please look here:

```
paste(getwd(), "/activity.csv", sep = "") -> reading

read.csv(reading) -> Q1

library(ggplot2)

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
```

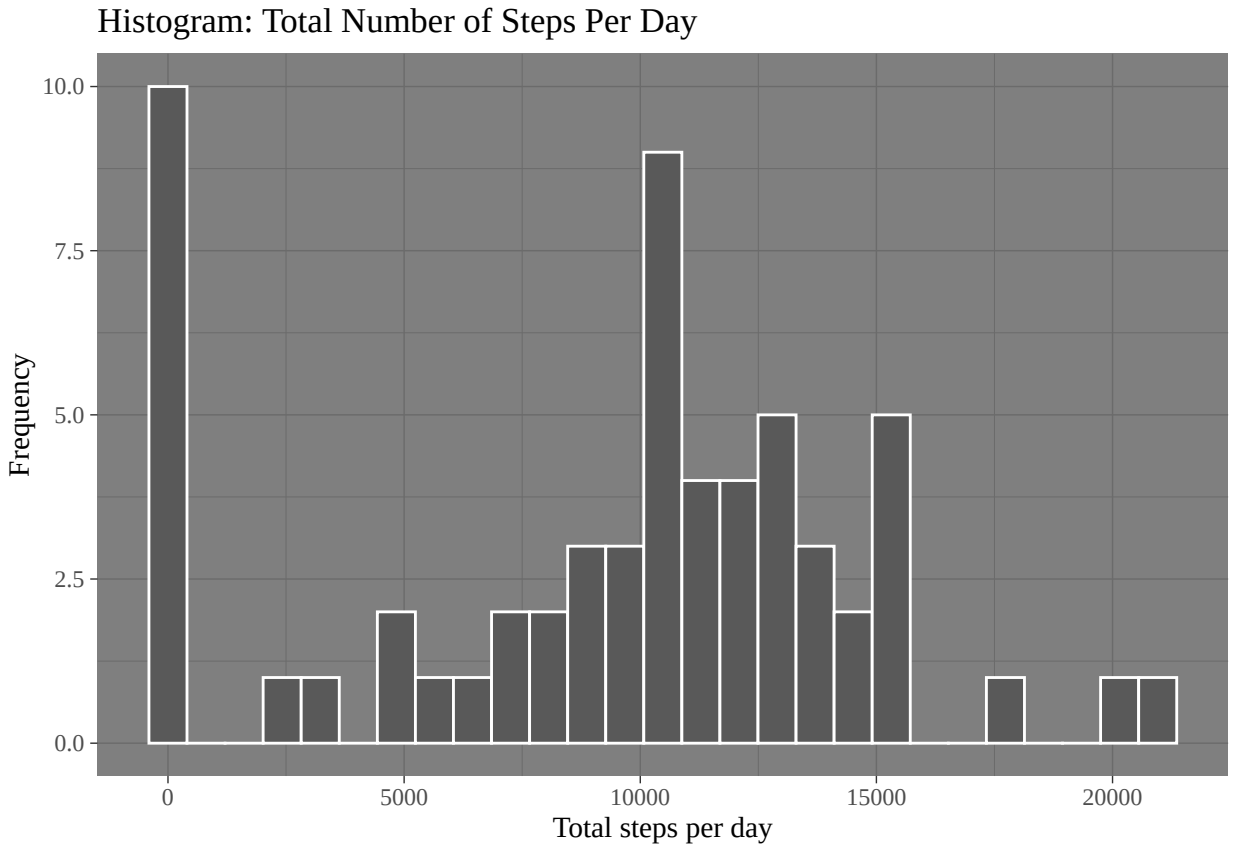
```
as.Date(Q1$date) -> Q1$date

tbl_df(Q1) -> Q1

Q1
```

```
## # A tibble: 17,568 x 3
##    steps date       interval
##    <int> <date>        <int>
##  1    NA 2012-10-01        0
```

```
##  2    NA 2012-10-01        5
##  3    NA 2012-10-01       10
##  4    NA 2012-10-01       15
##  5    NA 2012-10-01       20
##  6    NA 2012-10-01       25
##  7    NA 2012-10-01       30
##  8    NA 2012-10-01       35
##  9    NA 2012-10-01       40
## 10    NA 2012-10-01       45
## # ... with 17,558 more rows
```

The next part is to develop histogram to determine the total of steps per day, which is, admittedly, little challenging because the %>% tool is quite new. Despite this, the ggplot2 and dplyr both are in use to group and summary of summing steps from Q1 dataset and then develop the histogram.

```
Q1 %>% group_by(date) %>% summarise(Plot1 = sum(steps, na.rm = TRUE)) %>%
  ggplot(aes(x = Plot1)) + geom_histogram(binwidth = 806,
  na.rm = TRUE, color = "white") + ggtitle(
  "Histogram: Total Number of Steps Per Day") + xlab(
  "Total steps per day") +
  ylab("Frequency") + theme_dark(base_family = "Times")
```



The third part is to find mean and median of Q1 steps, which might be an easiest part to code.

```
Q1 %>% group_by(date) %>%
  summarise(mean(steps, na.rm = TRUE))
```

```
## # A tibble: 61 x 2
##    date       `mean(steps, na.rm = TRUE)`
```

```
##    <date>                          <dbl>
##  1 2012-10-01                      NaN
##  2 2012-10-02                       0.438
##  3 2012-10-03                      39.4
##  4 2012-10-04                      42.1
##  5 2012-10-05                      46.2
##  6 2012-10-06                      53.5
##  7 2012-10-07                      38.2
##  8 2012-10-08                      NaN
##  9 2012-10-09                      44.5
## 10 2012-10-10                      34.4
## # ... with 51 more rows
```

```
summarise(Q1, mean(steps, na.rm = TRUE))
```

```
## # A tibble: 1 x 1
##   `mean(steps, na.rm = TRUE)`
##                         <dbl>
## 1                        37.4
```

```
Q1 %>% group_by(date) %>%
  summarise(median(steps, na.rm = TRUE))
```

```
## # A tibble: 61 x 2
##    date       `median(steps, na.rm = TRUE)`
##    <date>                               <dbl>
##  1 2012-10-01                              NA
##  2 2012-10-02                               0
##  3 2012-10-03                               0
##  4 2012-10-04                               0
##  5 2012-10-05                               0
##  6 2012-10-06                               0
##  7 2012-10-07                               0
##  8 2012-10-08                              NA
##  9 2012-10-09                               0
## 10 2012-10-10                               0
## # ... with 51 more rows
```

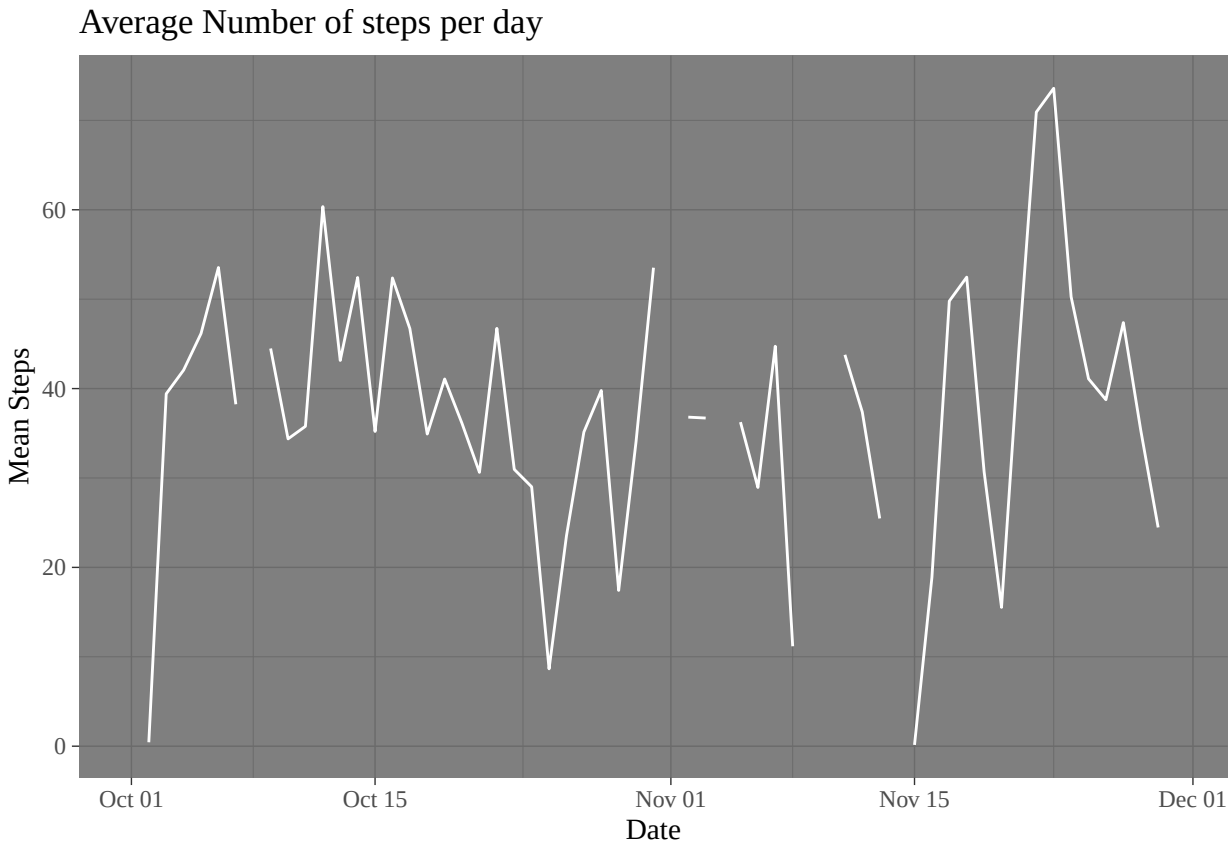
```
summarise(Q1, median(steps, na.rm = TRUE))
```

```
## # A tibble: 1 x 1
##   `median(steps, na.rm = TRUE)`
##                           <dbl>
## 1                             0
```

The fourth part is the plotting timeline of the average number of steps. In case if the disturbing warning indicates, it means that NA missing causes to create broken linear graph, which should be this way. To make graph smooth is possible by coding na.omit() at the beginning but since the instruction explains that NA missing should replace with mean of steps, this graph should not change.

```
Q1 %>% group_by(date) %>% summarise(Plot2 =
  mean(steps, na.rm = TRUE)) %>% ggplot(aes(
  x = date, y = Plot2))+ geom_line(color = "white") + labs(
  title = "Average Number of steps per day",
  x = "Date", y = "Mean Steps") + theme_dark(base_family =
  "Times")
```

```
## Warning: Removed 2 rows containing missing values (geom_path).
```



The fifth part is to find the maximum number of steps from within 5 minute interval through all of this dataset, which is fairly with simple coding.

```
Q1 %>% group_by(interval) %>% summarise(P = mean(steps,
  na.rm = TRUE)) %>% filter(P == max(P, na.rm = TRUE))
```

```
## # A tibble: 1 x 2
##   interval     P
##      <int> <dbl>
## 1      835  206.
```

As mentioned earlier, the instruction expects me to develop a strategy of coding the replacement for NA missing. This process requires a lot of thoughts and it is time-consuming but at least, it is successful.

```
# Locate the column containing with NA missing
Q1$steps %>% is.na() %>% as.numeric() %>% sum()
```

```
## [1] 2304
```

```
Q1$date %>% is.na() %>% as.numeric() %>% sum()
```

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

```
Q1$interval %>% is.na() %>% as.numeric() %>% sum()
```

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

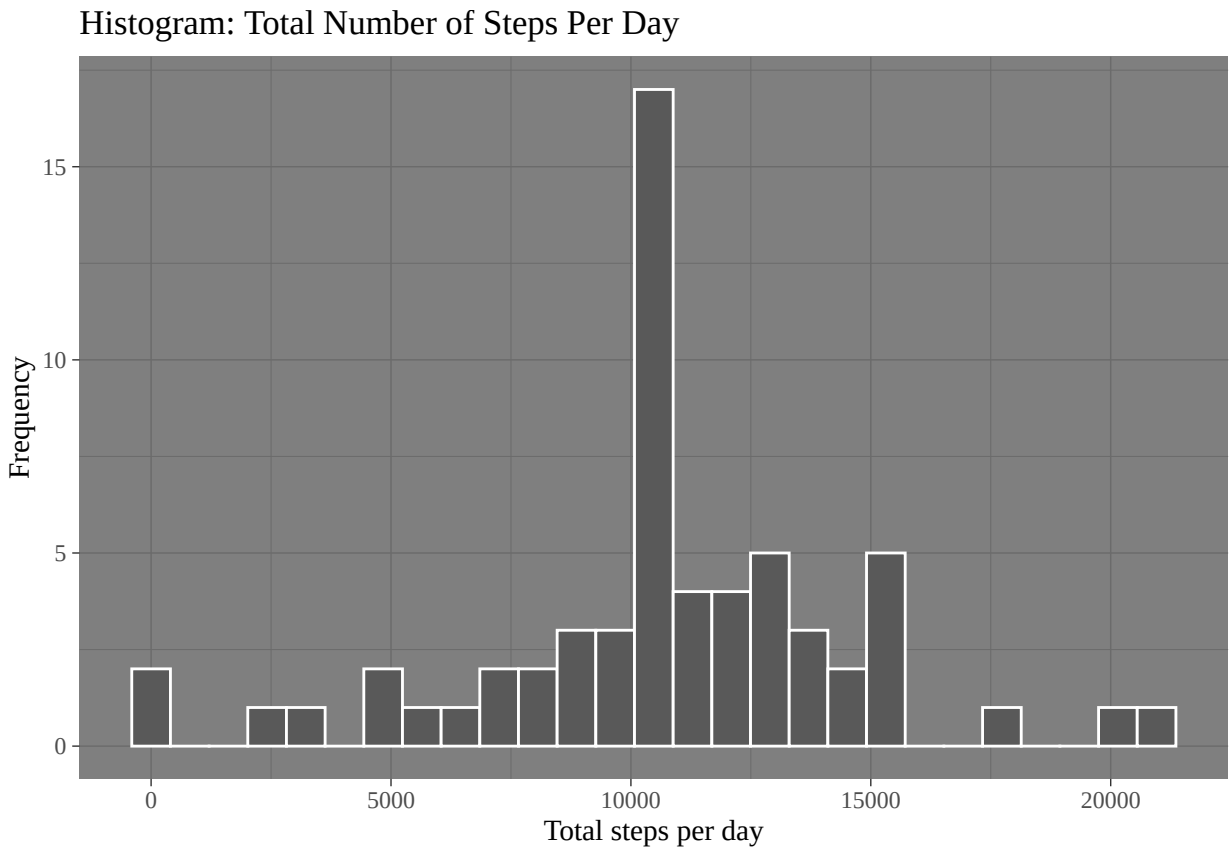
```
# Substitute NA with mean of steps on steps column only
arrange(Q1, steps) -> P
select(P, c(date, interval)) -> Keeper
```

```
select(P, steps) -> Editing
slice(Editing, 1:15264) -> p
x <- .001
while(x < 656700.2){
  add_row(p, steps = 37.3826) -> p
  as.numeric(lapply(p, sum)) -> x
}
cbind(p, Keeper) -> Q2
tbl_df(Q2) -> Q2
# The result shows that sum of NA missing is zero
Q2$steps %>% is.na() %>% as.numeric() %>% sum()
```

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

The seventh part is to develop histogram to determine the total of steps per day after the NA missing replaces. Also, the mean and median of steps display below this histogram because this information is necessary to reveal the change after NA missing replaces. Some changes are conspicuous, say, that in histogram value of frequency significantly declines in first few steps but significantly increases between 10000 and 12000 steps.

```
Q2 %>% group_by(date) %>% summarise(Plot1 = sum(steps)) %>%
  ggplot(aes(x = Plot1)) + geom_histogram(binwidth = 806, color =
  "white") + ggtitle(
  "Histogram: Total Number of Steps Per Day") + xlab(
  "Total steps per day") +
  ylab("Frequency") + theme_dark(base_family = "Times")
```




```
Q2 %>% group_by(date) %>%
  summarise(mean(steps))
```

```
## # A tibble: 61 x 2
##    date       `mean(steps)`
##    <date>             <dbl>
##  1 2012-10-01        37.4
##  2 2012-10-02         0.438
##  3 2012-10-03        39.4
##  4 2012-10-04        42.1
##  5 2012-10-05        46.2
##  6 2012-10-06        53.5
##  7 2012-10-07        38.2
##  8 2012-10-08        37.4
##  9 2012-10-09        44.5
## 10 2012-10-10        34.4
## # ... with 51 more rows
```

```
summarise(Q2, mean(steps))
```

```
## # A tibble: 1 x 1
##   `mean(steps)`
##           <dbl>
## 1          37.4
```

```
Q2 %>% group_by(date) %>%
  summarise(median(steps))
```

```
## # A tibble: 61 x 2
##    date       `median(steps)`
##    <date>               <dbl>
##  1 2012-10-01            37.4
##  2 2012-10-02             0
##  3 2012-10-03             0
##  4 2012-10-04             0
##  5 2012-10-05             0
##  6 2012-10-06             0
##  7 2012-10-07             0
##  8 2012-10-08            37.4
##  9 2012-10-09             0
## 10 2012-10-10             0
## # ... with 51 more rows
```

```
summarise(Q2, median(steps))
```

```
## # A tibble: 1 x 1
##   `median(steps)`
##             <dbl>
## 1               0
```

The final part is probably the one of most difficult things to code. It requires of me to install package named gridExtra, which allows to give more options by assembling two different plots into a panel. But at least, this coding is successful after many failures. The purpose of this panel is to compare how often

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
```

```
##     combine
```

```
Q2 -> Q3
Week <- sapply(Q3$date, function(x) {
  if (weekdays(x) == "Saturday" | weekdays(x) =="Sunday")
  {y <- "Weekend"} else
  {y <- "Weekday"}
  y
})
mutate(Q3, Week) -> Q3
filter(Q3, Week == "Weekend") -> Weekend
filter(Q3, Week == "Weekday") -> Weekday

Weekday %>% group_by(interval) %>% summarise(Plot2 =
  mean(steps)) %>% ggplot(aes(x = interval, y = Plot2)
  )+ geom_line(color = "cadetblue1") + labs(
  title = "Weekday",
  x = "Interval", y = "Mean Steps") + theme_dark(base_family =
  "Times") -> Pr
Weekend %>% group_by(interval) %>% summarise(Plot2 =
 mean(steps)) %>% ggplot(aes(x = interval, y = Plot2)
 )+ geom_line(color = "chartreuse") + labs(
 title = "Weekend",
 x = "Interval", y = "Mean Steps") + theme_dark(base_family =
 "Times") -> pR
grid.arrange(pR, Pr, nrow=2)
```