CourseProject2

Angela

9/15/2019

Loading and processing unimputed the data

Be sure to import the date column as an actual date.

```
activity <- read_csv("activity.zip",
    col_types = cols(date = col_date(format = "%Y-%m-%d"),
        interval = col_number(), steps = col_number()))
```

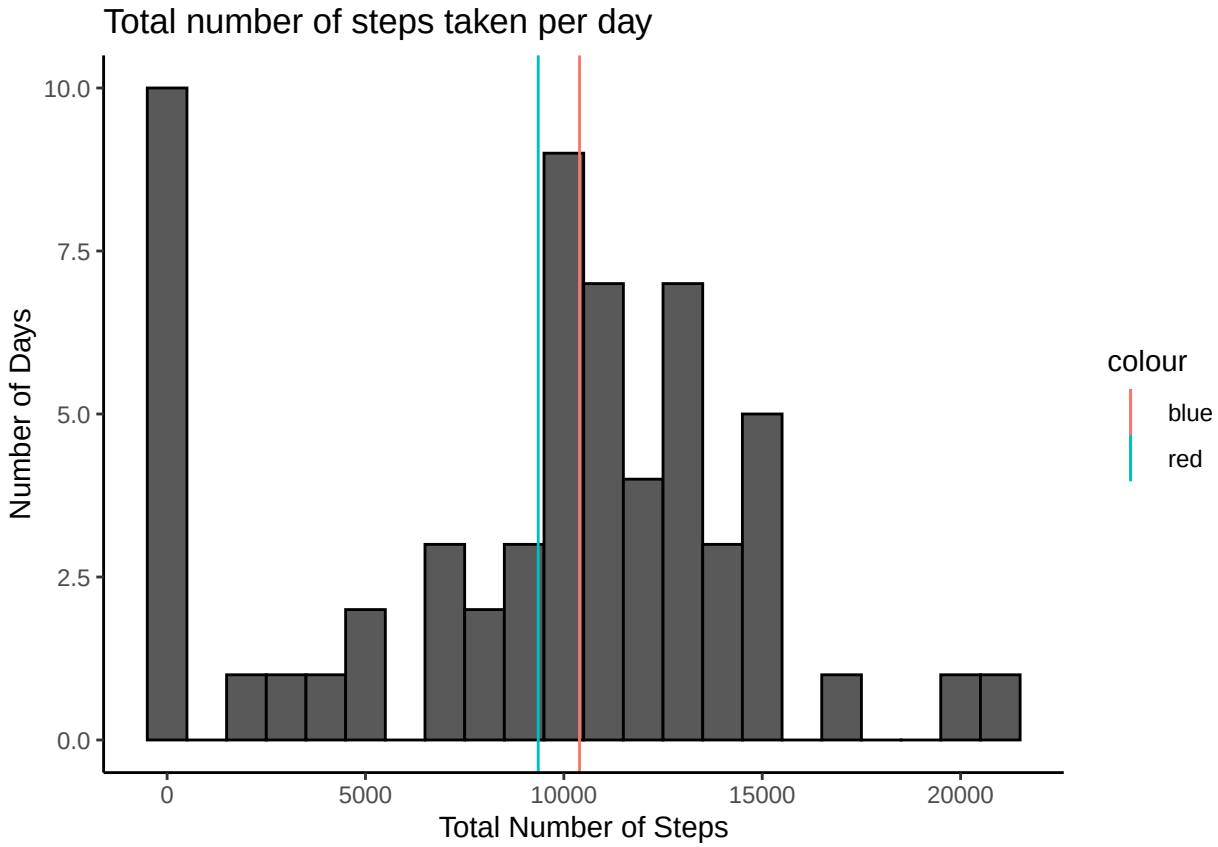
Calculating the number of steps taken per day

Use the mutate(), group_by() and summarise() functions from tidyverse to group the data by day and get the sum of the total number of steps.

```
steps_per_day <- activity %>%
    mutate(year = year(date), month = month(date), day = day(date)) %>%
    group_by(year, month, day) %>%
    summarise(total = sum(steps, na.rm = TRUE))
```

Now use ggplot package to create a histogram of the data. Be sure to indicate the mean and median values.

```
ggplot(steps_per_day, aes(total)) +
    geom_histogram(binwidth = 1000,
                   color = "black") +
    geom_vline(aes(xintercept = mean(total),
                   col = "red")) +
    geom_vline(aes(xintercept = median(total),
                   col = "blue")) +
    ggtitle("Total number of steps taken per day") +
    xlab("Total Number of Steps") +
    ylab("Number of Days") +
    theme_classic()
```




```
mean <- mean(steps_per_day$total, na.rm = TRUE)
med <- median(steps_per_day$total, na.rm = TRUE)
print(paste("The mean is ", mean, " while the median is ", med,"."))
```

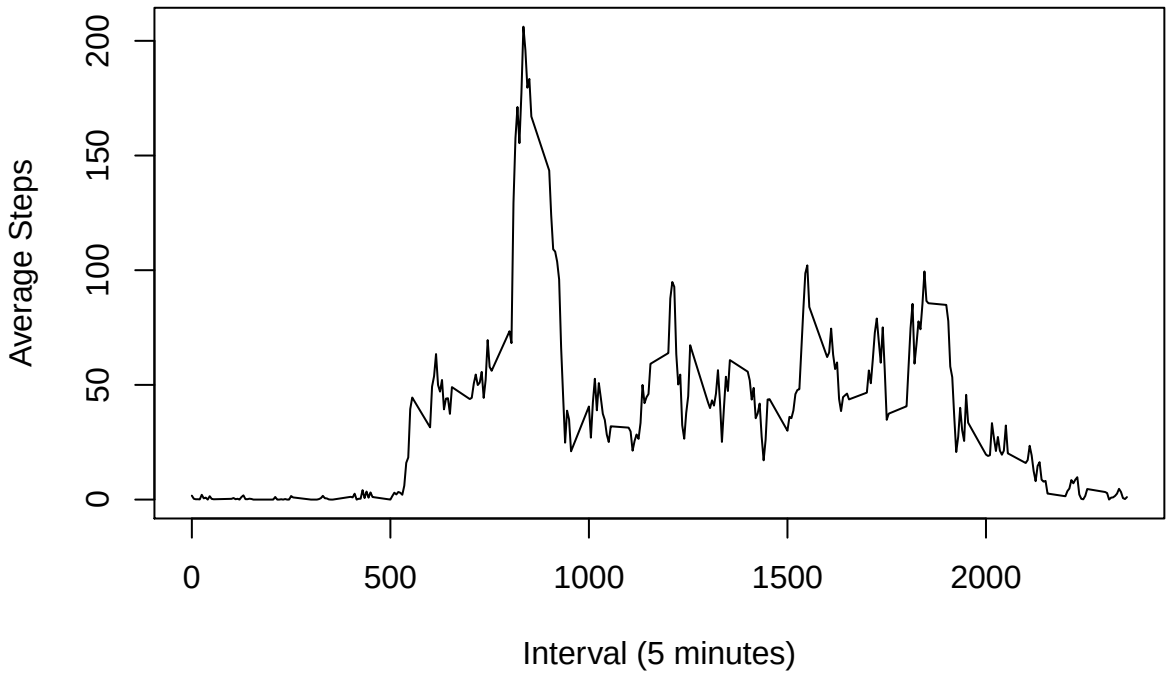
```
## [1] "The mean is  9354.22950819672  while the median is  10395 ."
```

Average Daily Acivity Pattern

Next, we want to make a time series line plot of the average number of steps taken at each time period.

```
steps_per_minute <- activity %>%
    group_by(interval) %>%
    summarise(avgsteps = mean(steps, na.rm = TRUE))
```

```
plot(x = steps_per_minute$interval,
     y = steps_per_minute$avgsteps,
     type = "l",
     ylab = "Average Steps",
     xlab = "Interval (5 minutes)")
```




```
top <- arrange(steps_per_minute, desc(avgsteps))
top1 <- as.numeric(top[1,1])

print(cat("The time interval when this person walked most during the day was ", top1, "."))
```

```
## The time interval when this person walked most during the day was  835 .NULL
```

Imputing missing values

Note that there are a number of days/intervals where there are missing values (coded as NA). The presence of missing days may introduce bias into some calculations or summaries of the data.

1. We will use the sum() and is.na() functions to calculate the total number of NA's in the steps column.

```
NAs <- sum(is.na(activity$steps))
print(NAs)
```

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

2-3. Devise a strategy for filling in all of the missing values in the dataset. The strategy does not need to be sophisticated and create a new dataset that is equal to the original dataset but with the missing data filled in.

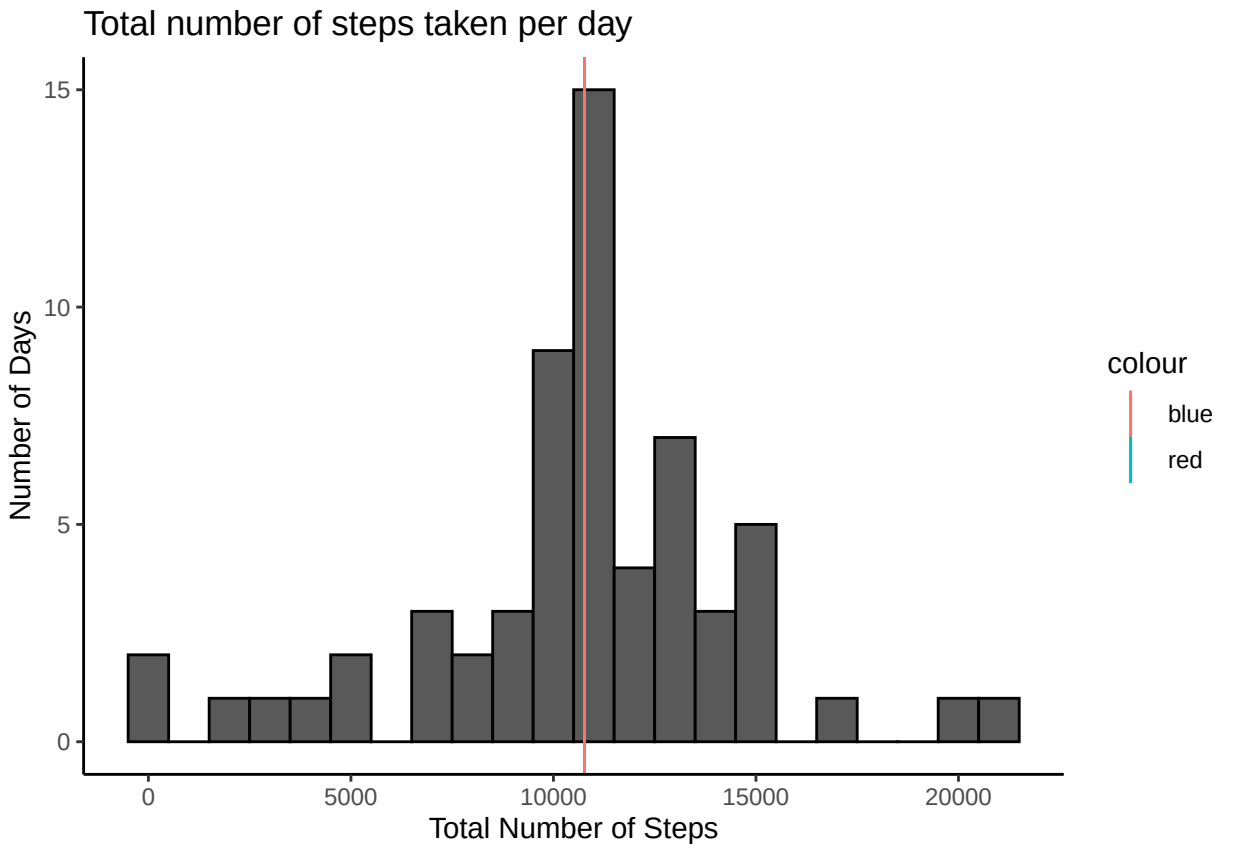
```
imputed_activity <- activity
imputed_activity$steps.mean <- ifelse(is.na(imputed_activity$steps), mean(imputed_activity$steps, na.rm
```

4. Make a histogram of the total number of steps taken each day and calculate and report the mean and median total number of steps taken per day. Do these values differ from the estimates from the first part of the assignment? What is the impact of imputing missing data on the estimates of the total daily number of steps?

```
imputed_steps_per_day <- imputed_activity %>%
    mutate(year = year(date), month = month(date), day = day(date)) %>%
    group_by(year, month, day) %>%
    summarise(total = sum(steps.mean, na.rm = TRUE))

ggplot(imputed_steps_per_day, aes(total)) +
```

```
    geom_histogram(binwidth = 1000,
                   color = "black") +
    geom_vline(aes(xintercept = mean(total),
                   col = "red")) +
    geom_vline(aes(xintercept = median(total),
                   col = "blue")) +
    ggtitle("Total number of steps taken per day") +
    xlab("Total Number of Steps") +
    ylab("Number of Days") +
    theme_classic()
```




```
imp_mean <- mean(imputed_steps_per_day$total, na.rm = TRUE)
imp_med <- median(imputed_steps_per_day$total, na.rm = TRUE)
print(paste("The mean is ", mean, " while the median is ", med,"."))
```

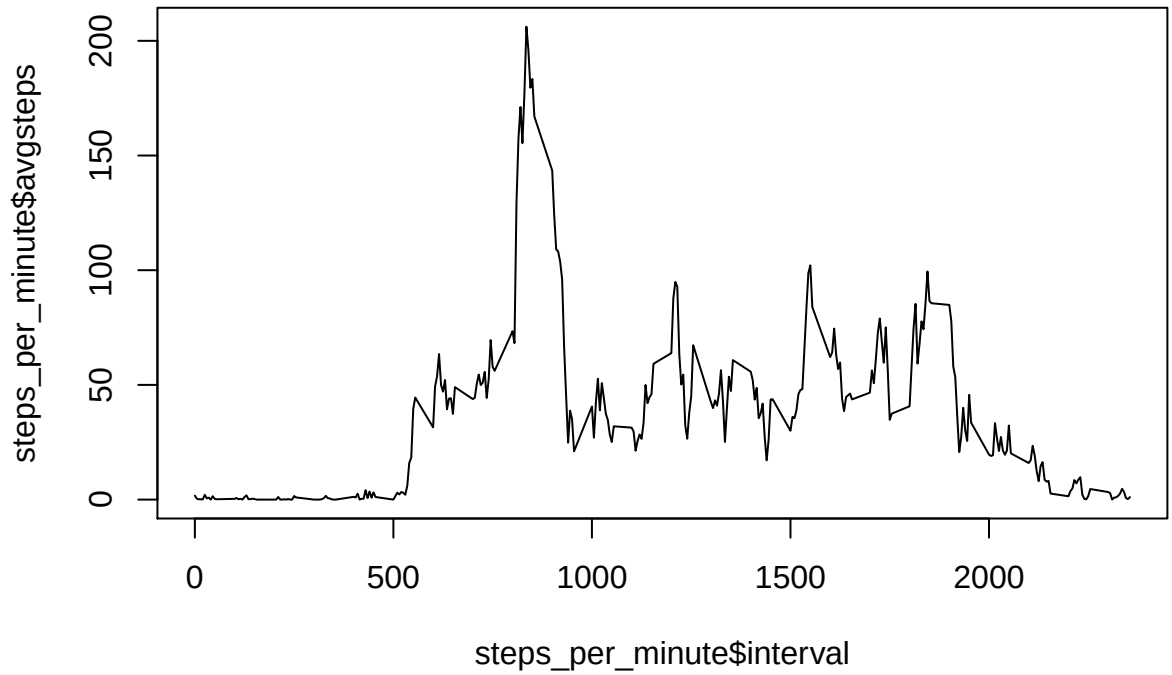
```
## [1] "The mean is  9354.22950819672  while the median is  10395 ."
```

```
print(cat("The time interval when this person walked most during the day was ", top1, "."))
```

```
## The time interval when this person walked most during the day was  835 .NULL
```

```
imputed_steps_per_minute <- imputed_activity %>%
    group_by(interval) %>%
    summarise(avgsteps = mean(steps.mean, na.rm = TRUE))

plot(x = steps_per_minute$interval, y = steps_per_minute$avgsteps, type = "l")
```




```
imp_top <- arrange(steps_per_minute, desc(avgsteps))
imp_top1 <- as.numeric(imp_top[1,1])

print(cat("The time interval when this person walked most during the day was ", imp_top1, "."))
```

```
## The time interval when this person walked most during the day was  835 .NULL
```

Imputing mean values does not change the results.

Are there differences in activity patterns between weekdays and weekends?

For this part the weekdays() function may be of some help here. Use the dataset with the filled-in missing values for this part.

Create a new factor variable in the dataset with two levels – "weekday" and "weekend" indicating whether a given date is a weekday or weekend day. Make a panel plot containing a time series plot (i.e. type="l") of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all weekday days or weekend days (y-axis). See the README file in the GitHub repository to see an example of what this plot should look like using simulated data.

Use the weekday() function to determine what day of the week it is then use ifelse() to make a new Weekend and Weekday column. Create a graph that facets weekends and weekdays.

```
imputed_activity$daytype <- weekdays(imputed_activity$date)

weekend_days <- c("Saturday", "Sunday")

imputed_activity$weekendorweekday <- ifelse(imputed_activity$daytype == weekend_days,
                                            "Weekend",
                                            "Weekday")
imputed_activity$weekendorweekday <- as.factor(imputed_activity$weekendorweekday)

imputed_steps_per_minute_bydaytype <- imputed_activity %>%
    group_by(interval, weekendorweekday) %>%
```

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

ggplot(imputed_steps_per_minute_bydaytype,
       aes(interval, avgsteps)) +
    geom_line() +
    facet_grid(weekendorweekday~.)
```