reproducible_res_proj1

Larry YE

26 July 2016

Introduction

This assignment makes use of data from a personal activity monitoring device. This device collects data at 5 minute intervals throughout the day. The data consists of two months of data from an anonymous individual collected during the months of October and November, 2012 and include the number of steps taken in 5 minute intervals each day.

This document presents the results from Project Assignment 1 in the Coursera course Reproducible Research, written in a single R markdown document that can be processed by knitr and transformed into an HTML file.

The environment set up

In this document code will be represented to show how the results have been achieved. Set the default of echo to be true throughout the document:

```
setwd("C:/00DS/R/5ReproducibleResearch")

ipak <- function(pkg){
  new.pkg <- pkg[!(pkg %in% installed.packages()[, "Package"])]
  if(length(new.pkg))
      install.packages(new.pkg, dependencies =  TRUE)
    sapply(pkg,require,character.only = TRUE)
}

packages<-c("knitr","dplyr","lubridate","ggplot2")
ipak(packages)
```

```
## Loading required package: knitr
```

```
## Loading required package: 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

## Loading required package: lubridate

##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date

## Loading required package: ggplot2

##     knitr     dplyr lubridate   ggplot2
##      TRUE      TRUE      TRUE      TRUE

library(knitr); library(dplyr); library(ggplot2)
opts_chunk$set(echo = TRUE)
```

Loading and preprocessing the data

Code for reading in the dataset and/or processing the data Show the code that is needed to: Load the data (i.e. read.csv()); Process/transform the data (if necessary) into a format suitable for your analysis The data is loaded using the read.csv(). NOTE: It is assumed that you have already downloaded the activity.csv and saved it in your working directory. If not, please download the code here, unzip it and save it to your working directory. Then, change the date into dateformat using lubridate.

```
data <- read.csv("activity.csv", header = TRUE, sep = ',', colClasses =
c("numeric", "character", "integer"))
        data$date <- ymd(data$date)
str(data); head(data)

## 'data.frame':    17568 obs. of  3 variables:
##  $ steps   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ date    : Date, format: "2012-10-01" "2012-10-01" ...
##  $ interval: int  0 5 10 15 20 25 30 35 40 45 ...

##   steps       date interval
## 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
```

What is mean total number of steps taken per day?

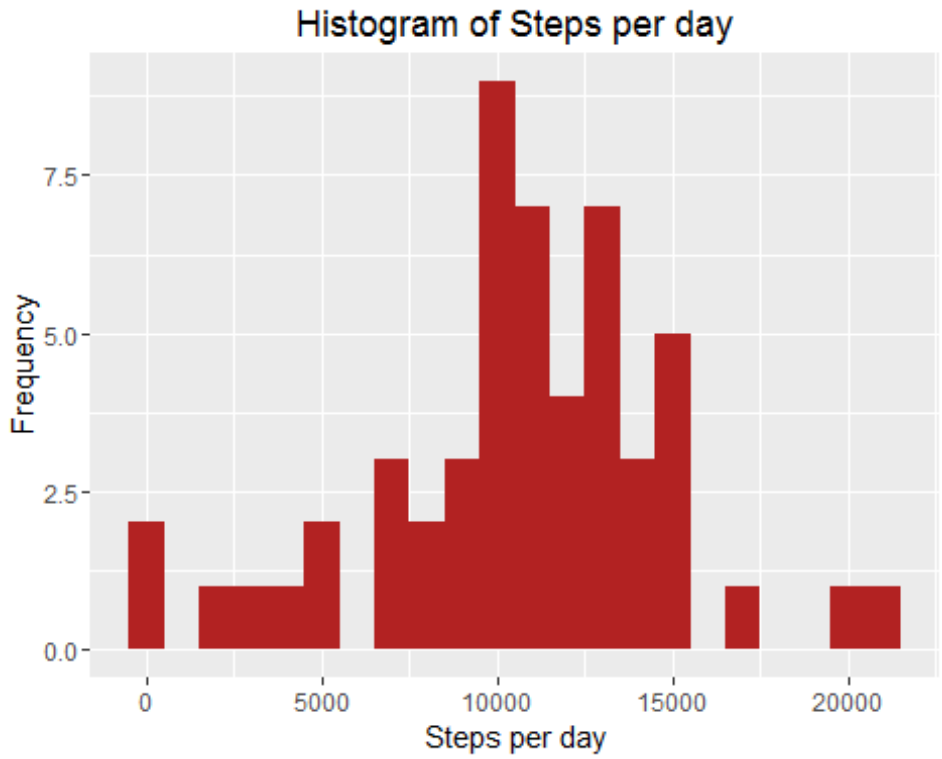
For this part of the assignment, you can ignore the missing values in the dataset.

1.Calculate the total number of steps taken per day 2.If you do not understand the difference between a histogram and a barplot, research the difference between them. Make a histogram of the total number of steps taken each day 3.Calculate and report the mean and median of the total number of steps taken per day

```
steps <- data %>%
  filter(!is.na(steps)) %>%
  group_by(date) %>%
  summarize(steps = sum(steps)) %>%
  print
```

```
## Source: local data frame [53 x 2]
##
##          date steps
##        <date> <dbl>
## 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
## ..        ...   ...
```

```
ggplot(steps, aes(x = steps)) +
  geom_histogram(fill = "firebrick", binwidth = 1000) +
  labs(title = "Histogram of Steps per day", x = "Steps per day", y =
"Frequency")
```




```
mean_steps <- mean(steps$steps, na.rm = TRUE)
median_steps <- median(steps$steps, na.rm = TRUE)

mean_steps; median_steps
```

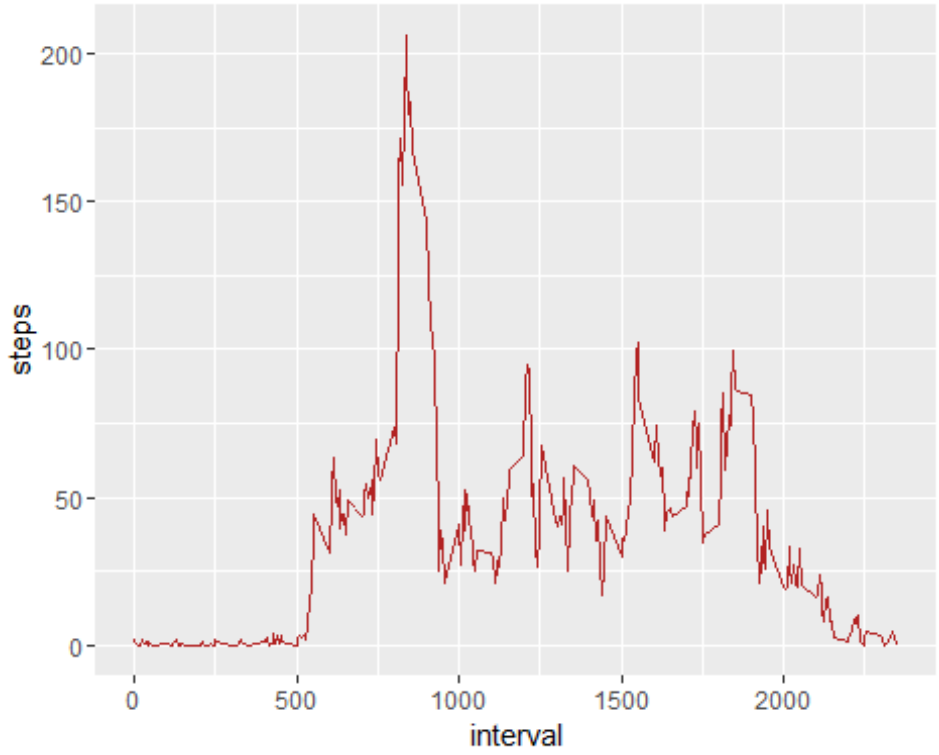
```
## [1] 10766.19
```

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

What is the average daily activity pattern?

1.Make 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 days (y-axis)

```
interval <- data %>%
  filter(!is.na(steps)) %>%
  group_by(interval) %>%
  summarize(steps = mean(steps))

ggplot(interval, aes(x=interval, y=steps)) +
  geom_line(color = "firebrick")
```



2.Which 5-minute interval, on average across all the days in the dataset, contains the maximum number of steps?

```
interval[which.max(interval$steps),]
```

```
## Source: local data frame [1 x 2]
##
##   interval    steps
##      <int>    <dbl>
## 1      835 206.1698
```

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.Calculate and report the total number of missing values in the dataset (i.e. the total number of rows with NAs)

```
sum(is.na(data$steps))
```

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

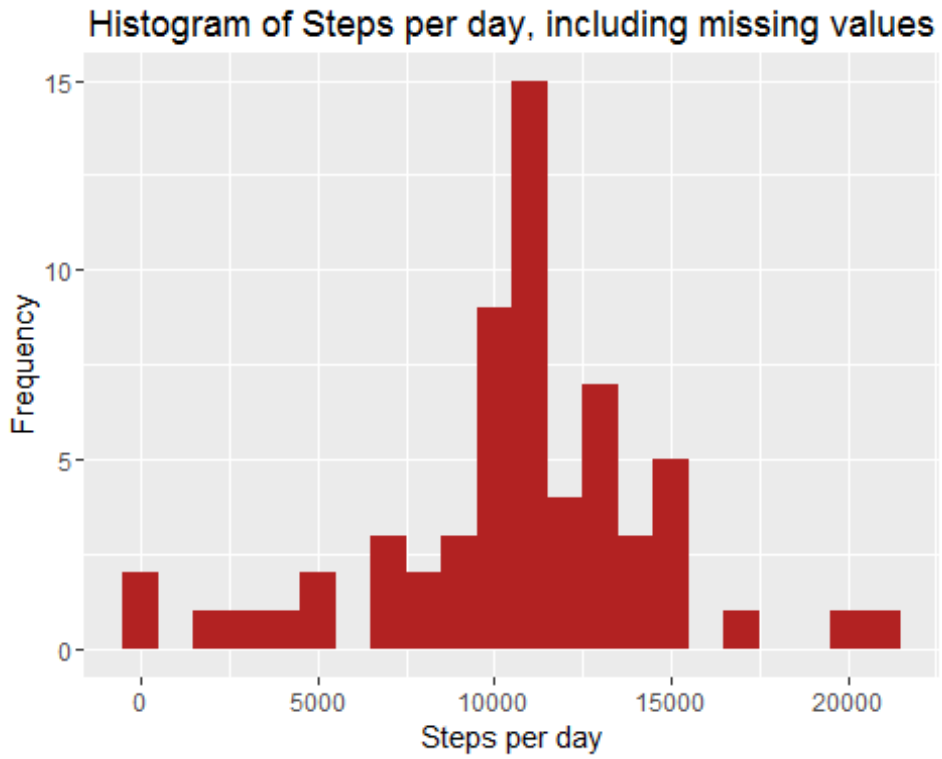
2.Devise a strategy for filling in all of the missing values in the dataset. The strategy does not need to be sophisticated. For example, you could use the mean/median for that day, or

the mean for that 5-minute interval, etc. 3.Create a new dataset that is equal to the original dataset but with the missing data filled in.

```
data_full <- data
nas <- is.na(data_full$steps)
avg_interval <- tapply(data_full$steps, data_full$interval, mean, na.rm=TRUE,
simplify=TRUE)
data_full$steps[nas] <- avg_interval[as.character(data_full$interval[nas])]

sum(is.na(data_full$steps))

## [1] 0
```

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?

```
steps_full <- data_full %>%
  filter(!is.na(steps)) %>%
  group_by(date) %>%
  summarize(steps = sum(steps)) %>%
  print

## Source: local data frame [61 x 2]
##
##          date    steps
##        <date>    <dbl>
## 1  2012-10-01 10766.19
## 2  2012-10-02   126.00
## 3  2012-10-03 11352.00
## 4  2012-10-04 12116.00
## 5  2012-10-05 13294.00
## 6  2012-10-06 15420.00
## 7  2012-10-07 11015.00
## 8  2012-10-08 10766.19
## 9  2012-10-09 12811.00
## 10 2012-10-10  9900.00
## ..        ...      ...

ggplot(steps_full, aes(x = steps)) +
  geom_histogram(fill = "firebrick", binwidth = 1000) +
  labs(title = "Histogram of Steps per day, including missing values", x =
"Steps per day", y = "Frequency")
```




```
mean_steps_full <- mean(steps_full$steps, na.rm = TRUE)
median_steps_full <- median(steps_full$steps, na.rm = TRUE)
mean_steps_full
```

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

```
median_steps_full
```

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

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.

1.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.

```
data_full <- mutate(data_full, weektype = ifelse(weekdays(data_full$date) ==
"Saturday" | weekdays(data_full$date) == "Sunday", "weekend", "weekday"))
data_full$weektype <- as.factor(data_full$weektype)
head(data_full)
```

```
##       steps       date interval weektype
## 1 1.7169811 2012-10-01        0  weekday
```

```
## 2 0.3396226 2012-10-01        5  weekday
## 3 0.1320755 2012-10-01       10  weekday
## 4 0.1509434 2012-10-01       15  weekday
## 5 0.0754717 2012-10-01       20  weekday
## 6 2.0943396 2012-10-01       25  weekday
```

2.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.

```
interval_full <- data_full %>%
  group_by(interval, weektype) %>%
  summarise(steps = mean(steps))
s <- ggplot(interval_full, aes(x=interval, y=steps, color = weektype)) +
  geom_line() +
  facet_wrap(~weektype, ncol = 1, nrow=2)
print(s)
```