Activity Analysis

Stephen Hobbs

April 13, 2016

It is now possible to collect a large amount of data about personal movement using activity monitoring devices such as a Fitbit, Nike Fuelband, or Jawbone Up. These type of devices are part of the “quantified self” movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. But these data remain under-utilized both because the raw data are hard to obtain and there is a lack of statistical methods and software for processing and interpreting the data.

This research analyzes the amount of steps taken in five minute intervals throughout a week. It examines mean and median number of steps and asks the question, "When were more average steps taken: weekday or weekend?"

To begin download the data set from this url: <https://www.coursera.org/learn/reproducible-research/peer/gYyPt/course-project-1> Click on the hyperlink, "Activity monitoring data." Download and save to your hard drive. The variables included in this dataset are:

steps: Number of steps taking in a 5-minute interval (missing values are coded as NA)  
date: The date on which the measurement was taken in YYYY-MM-DD format  
interval: Identifier for the 5-minute interval in which measurement was taken

The dataset is stored in a comma-separated-value (CSV) file and there are a total of 17,568 observations in this dataset. Loading and preprocessing the data, import .csv file. Assign the variable, data, to the file.

Load R package to be used later.

setwd("/Users/stephenhobbs1/Data science/5-Reproducible Research")  
data <- read.csv("activity.csv", stringsAsFactors=FALSE, header = TRUE)  
library(plyr)  
library(lattice)

# What is total number of steps taken?

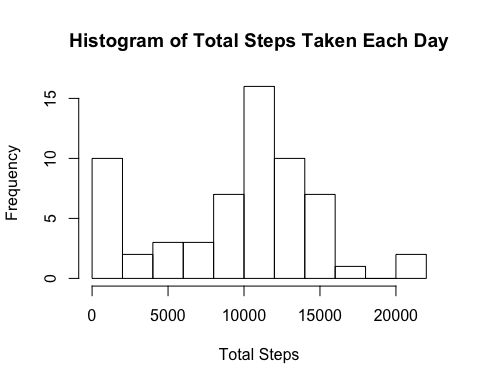
Summarize the data by date and number of steps using function ddpy from plyr package. Do not remove "NA" values yet. Assign this data set to "tmp" variable. "tmp" gives you the total number of steps by date.

#Summarize dataframe by date, total steps  
tmp<-ddply(data,.(date),summarise, totalSteps=sum(steps, na.rm=TRUE))   
# Sum of total steps, 570,608 steps.  
sum(tmp$totalSteps)

## [1] 570608

Here is the histogram.

hist(tmp$totalSteps, breaks = 10, xlab = "Total Steps",  
 main = "Histogram of Total Steps Taken Each Day")



Next is the code of the average and median steps.

# Average step is 9354.23  
mean(tmp$totalSteps, na.rm=TRUE)

## [1] 9354.23

# Median step is 10,395  
median(tmp$totalSteps, na.rm=TRUE)

## [1] 10395

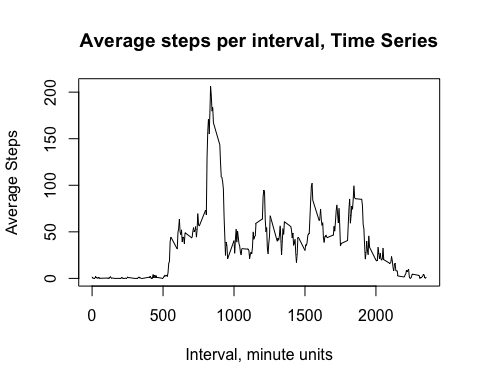
# What is the average daily activity pattern?

Use the plyr package, ddply function, to get the average steps per time interval and plot. Assign new variable, tmp2, to this data frame.

#Summarize date set by date, average steps/interval  
tmp2<-ddply(data,.(interval),summarise, AverageSteps=mean(steps, na.rm=TRUE))   
# Average number of steps to use in imputing NA values  
mean(tmp2$AverageSteps)

## [1] 37.3826

# Plot average steps per interval using a line as the type.   
plot(tmp2, type="l", main = "Average steps per interval, Time Series", ylab="Average Steps", xlab="Interval, minute units")



# Which interval had the most steps? Run this code to discover.

#Answer is 206.17 steps at 0835.  
tmp2[which.max(tmp2$AverageSteps),]

## interval AverageSteps  
## 104 835 206.1698

# Imputing missing values

I took the average of the steps per interval and substituted the average for all NA values. Number of NA values is 2304. Here is the code.

# Code shows number of "NA" values, or 2304 "NA" values.  
sum(is.na(data))

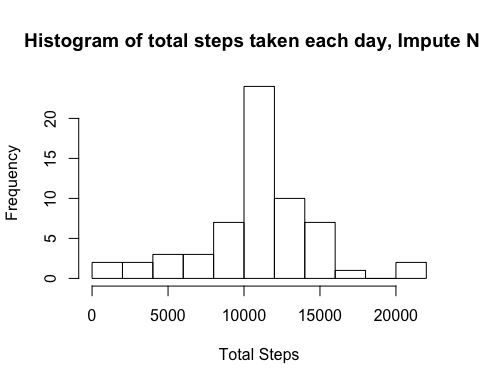
## [1] 2304

Then we want to group data on date and total steps. First, we calculate the average of the 5 minute interval averages. This will then be the number we use to impute the value for NA. That number is 37.3826.

#Replacing missing NA value with average of average of 5 min values. Merge original data set with tmp2, which shows average.  
datanew<-merge(data, tmp2, by="interval", sort=TRUE)   
index <- is.na(datanew$steps) #Returns logical vector, true = missing value  
#Here we impute. Take out rows of missing values and replace with #AverageSteps where there was NA, missing value  
datanew$steps[index] <- datanew$AverageSteps[index]   
#Group on date and total steps  
tmp3<-ddply(datanew, .(date), summarise, totalsteps=sum(steps, na.rm = TRUE))

Below is the code for a histogram of averages with using imputed average for missing values.

hist(tmp3$totalsteps, breaks=10, main = "Histogram of total steps taken each day, Impute NA", xlab="Total Steps") #Histogram with imputed missing values



Below is the code for mean and median steps with imputed average replacing missing value. The mean and median wer 10,766.19 steps. The average and median using imputed average in place of NA differ from the average and median with NA in the data set, 9534 and 10,395 respectively.

# Imputed mean of steps, NA removed. Average steps = 10,766.19 steps  
mean(tmp3$totalsteps)

## [1] 10766.19

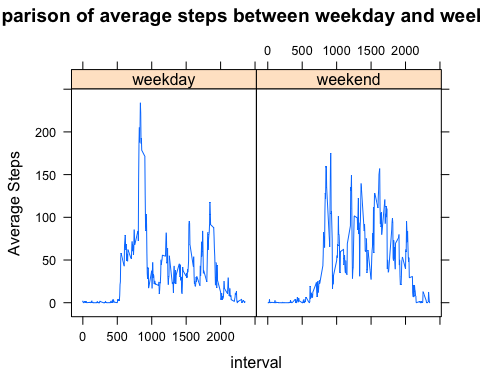
# Imputed median of steps, NA removed. Median steps = 10,766.19 steps  
median(tmp3$totalsteps)

## [1] 10766.19

# Differences between weekday and weekend activity

We separate steps of weekdays from weekends.

# Convert dates into weekdays and weekends  
tmp4<-transform(data, date = strptime(date, format="%Y-%m-%d"))  
#Use weekdays() function to classify day. I commented the next line to reduce the file size; it was too big for Git   
#Hub.  
#weekdays(tmp4$date) %in% c("Saturday", "Sunday")   
# Set logical if day is weekday or weekend. I commented the next line to reduce the file size; it was too big for #Git Hub.  
#factor(weekdays(tmp4$date) %in% c("Saturday", "Sunday"), level=c(TRUE, FALSE), labels=c("weekend", "weekday"))  
# Assign factor variable to day. I commented the next line to reduce the file size; it was too big for Git  
#Hub.  
day1<-factor(weekdays(tmp4$date) %in% c("Saturday", "Sunday"), level=c(TRUE, FALSE), labels=c("weekend", "weekday"))  
#Create new column, weekend, weekend using mutate from plyr  
tmp4<-mutate(tmp4, day=day1)  
#Plot weekend and weekday averages; change day from POSIXlt format to character format because #ddply does not work on posixlt; it only works on charactoer & number vector.  
tmp4<-transform(tmp4, day1=as.character((day1)))   
tmp5<-ddply(tmp4,.(interval,day1), summarize, averagesteps=mean(steps, na.rm = TRUE))  
#Plot weekday and weekend average steps per interval using line graph.  
  
xyplot(averagesteps~interval|day1, data = tmp5,   
 ylab="Average Steps",   
 type = "l",  
 main="Comparison of average steps between weekday and weekend")



# Weekend average is higher on average but maximum value was highest during weekday.