Reproducible Research Project 1

Ravinendra Pratap

10 December 2018

## R Markdown

# Reproducible Research: Peer Assessment 1

## Loading and preprocessing the data

getwd()

## [1] "D:/LND/COURSERA\_DATA\_SCIENCE/COURSERA\_05\_Reproducible Research/WEEK2\_05RR\_Markdown\_knitr/Assignment"

setwd ("D:/LND/COURSERA\_DATA\_SCIENCE/COURSERA\_05\_Reproducible Research/WEEK2\_05RR\_Markdown\_knitr/Assignment")

Loading and preprocessing the data

install.packages(“ggplot2”) install.packages(“dplyr”) install.packages(“chron”)

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

library (chron)

##### 1. Load the data (i.e. read.csv())

##### Downloading zip file if it doesn’t already exist in the workspace

path <- getwd()  
download.file(url = "https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip"  
 , destfile = paste(path, "dataFiles.zip", sep = "/"))  
unzip(zipfile = "dataFiles.zip")

Clear the workspace load raw activity data

rm(list=ls())  
activity\_raw <- read.csv("activity.csv", stringsAsFactors=FALSE)

## Process/transform the data suitable for analysis

### Transform the date attribute to an actual date format

activity\_raw$date <- as.POSIXct(activity\_raw$date, format="%Y-%m-%d")  
activity\_raw <- data.frame(date=activity\_raw$date,   
 weekday=tolower(weekdays(activity\_raw$date)),   
 steps=activity\_raw$steps,   
 interval=activity\_raw$interval)

Compute the day type (weekend or weekday)

activity\_raw <- cbind(activity\_raw,   
 daytype=ifelse(activity\_raw$weekday == "saturday" |   
 activity\_raw$weekday == "sunday", "weekend",   
 "weekday"))  
  
activity <- data.frame(date=activity\_raw$date,   
 weekday=activity\_raw$weekday,   
 daytype=activity\_raw$daytype,   
 interval=activity\_raw$interval,  
 steps=activity\_raw$steps)  
  
rm(activity\_raw)

Checking activity frame

dim(activity)

## [1] 17568 5

head(activity)

## date weekday daytype interval steps  
## 1 2012-10-01 monday weekday 0 NA  
## 2 2012-10-01 monday weekday 5 NA  
## 3 2012-10-01 monday weekday 10 NA  
## 4 2012-10-01 monday weekday 15 NA  
## 5 2012-10-01 monday weekday 20 NA  
## 6 2012-10-01 monday weekday 25 NA

str(activity)

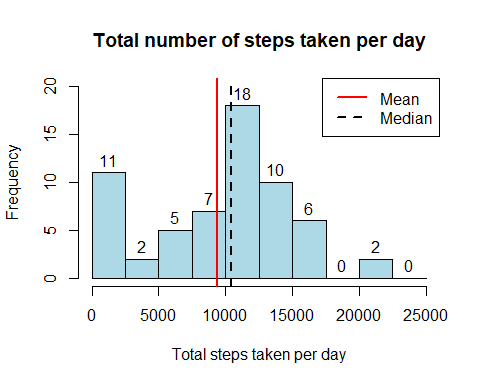
## 'data.frame': 17568 obs. of 5 variables:  
## $ date : POSIXct, format: "2012-10-01" "2012-10-01" ...  
## $ weekday : Factor w/ 7 levels "friday","monday",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ daytype : Factor w/ 2 levels "weekday","weekend": 1 1 1 1 1 1 1 1 1 1 ...  
## $ interval: int 0 5 10 15 20 25 30 35 40 45 ...  
## $ steps : int NA NA NA NA NA NA NA NA NA NA ...

summary(activity)

## date weekday daytype interval   
## Min. :2012-10-01 friday :2592 weekday:12960 Min. : 0.0   
## 1st Qu.:2012-10-16 monday :2592 weekend: 4608 1st Qu.: 588.8   
## Median :2012-10-31 saturday :2304 Median :1177.5   
## Mean :2012-10-31 sunday :2304 Mean :1177.5   
## 3rd Qu.:2012-11-15 thursday :2592 3rd Qu.:1766.2   
## Max. :2012-11-30 tuesday :2592 Max. :2355.0   
## wednesday:2592   
## steps   
## Min. : 0.00   
## 1st Qu.: 0.00   
## Median : 0.00   
## Mean : 37.38   
## 3rd Qu.: 12.00   
## Max. :806.00   
## NA's :2304

##### 1. Make a histogram of the total number of steps taken each day

activity\_total\_steps <- with(activity, aggregate(steps, by = list(date), FUN = sum, na.rm = TRUE))  
names(activity\_total\_steps) <- c("date", "steps")  
hist(activity\_total\_steps$steps, main = "Total number of steps taken per day", xlab = "Total steps taken per day", col = "lightblue", ylim = c(0,20), breaks = seq(0,25000, by=2500), labels=TRUE)  
abline(v = mean(activity\_total\_steps$steps), lty = 1, lwd = 2, col = "red")  
abline(v = median(activity\_total\_steps$steps), lty = 2, lwd = 2, col = "black")  
legend(x = "topright", c("Mean", "Median"), col = c("red", "black"),   
 lty = c(1, 2), lwd = c(2, 2))



##Mean  
mean(activity\_total\_steps$steps)

## [1] 9354.23

##Median  
median(activity\_total\_steps$steps)

## [1] 10395

summary(activity\_total\_steps$steps)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0 6778 10395 9354 12811 21194

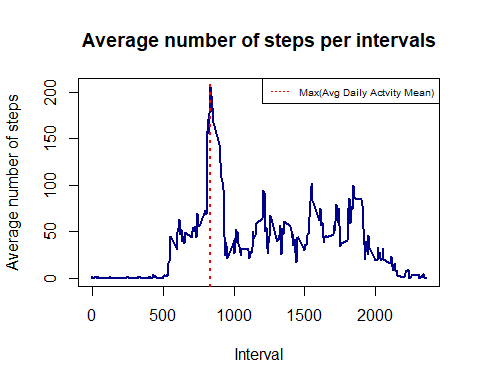
## What is the average daily activity pattern?

## Excludes Missing Values“NA” using na.rm=TRUE

average\_daily\_activity <- aggregate(activity$steps, by=list(activity$interval), FUN=mean, na.rm=TRUE)  
names(average\_daily\_activity) <- c("interval", "mean")  
  
plot(average\_daily\_activity$interval, average\_daily\_activity$mean, type = "l", col="darkblue", lwd = 2, xlab="Interval", ylab="Average number of steps", main="Average number of steps per intervals")  
average\_daily\_activity[which.max(average\_daily\_activity$mean), ]$interval

## [1] 835

abline(v = average\_daily\_activity[which.max(average\_daily\_activity$mean), ]$interval, lty = 3, lwd = 2, col = "red")  
legend(x = "topright", c("Max(Avg Daily Actvity Mean)"), col = c("red"),lty = c(3), cex=0.65)



## Max Average   
## average\_daily\_activity[which.max(average\_daily\_activity$mean), [1]  
  
average\_daily\_activity[which.max(average\_daily\_activity$mean), ]$interval

## [1] 835

## Maximum Average Number of Steps  
## average\_daily\_activity[which.max(average\_daily\_activity$mean), ][2]  
  
average\_daily\_activity[which.max(average\_daily\_activity$mean), ]$mean

## [1] 206.1698

# Split into two sets: complete and missing.

activity.missing <- activity[is.na(activity$steps),]  
activity.complete<-activity[complete.cases(activity),]  
  
NA\_count <- sum(is.na(activity$steps))  
NA\_pos <- which(is.na(activity$steps))  
mean\_vec <- rep(mean(activity$steps, na.rm=TRUE), times=length(NA\_pos))  
activity.complete[NA\_pos, "steps"] <- mean\_vec  
head(activity.complete)

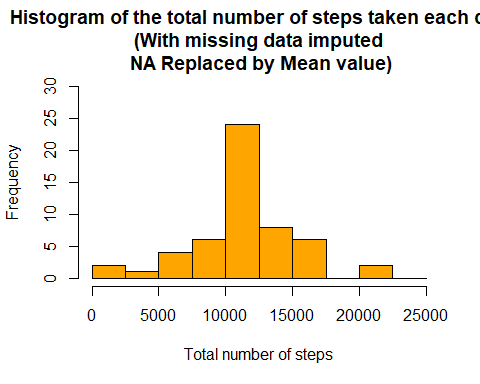
## date weekday daytype interval steps  
## 289 2012-10-02 tuesday weekday 0 37.3826  
## 290 2012-10-02 tuesday weekday 5 37.3826  
## 291 2012-10-02 tuesday weekday 10 37.3826  
## 292 2012-10-02 tuesday weekday 15 37.3826  
## 293 2012-10-02 tuesday weekday 20 37.3826  
## 294 2012-10-02 tuesday weekday 25 37.3826

# Compute the total number of steps each day (NA values removed)

sum\_data <- aggregate(activity.complete$steps, by=list(activity.complete$date), FUN=sum)  
  
## Rename the attributes  
names(sum\_data) <- c("date", "total")

# Compute the histogram of the total number of steps each day

hist(sum\_data$total,   
 breaks=seq(from=0, to=25000, by=2500),  
 col="orange",   
 xlab="Total number of steps",   
 ylim=c(0, 30),   
 main="Histogram of the total number of steps taken each day\n(With missing data imputed\n NA Replaced by Mean value)")



## Mean  
 mean(sum\_data$total)

## [1] 11126.8

## Median  
median(sum\_data$total)

## [1] 10766.19

## Clear the workspace  
rm(sum\_data)  
  
## Load the lattice graphical library---  
library(lattice)

# Compute the average number of steps taken, averaged across all daytype variable

head(activity.complete)

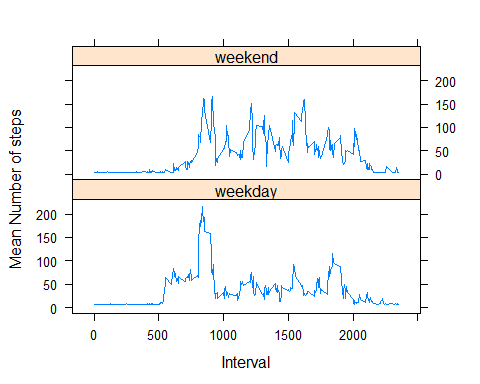
## date weekday daytype interval steps  
## 289 2012-10-02 tuesday weekday 0 37.3826  
## 290 2012-10-02 tuesday weekday 5 37.3826  
## 291 2012-10-02 tuesday weekday 10 37.3826  
## 292 2012-10-02 tuesday weekday 15 37.3826  
## 293 2012-10-02 tuesday weekday 20 37.3826  
## 294 2012-10-02 tuesday weekday 25 37.3826

activity.complete.daytype <- aggregate(steps ~ daytype+interval, data=activity.complete, FUN=mean)  
head(activity.complete.daytype)

## daytype interval steps  
## 1 weekday 0 7.212708  
## 2 weekend 0 2.670186  
## 3 weekday 5 5.751169  
## 4 weekend 5 2.670186  
## 5 weekday 10 5.751169  
## 6 weekend 10 2.670186

# Compute the time serie plot

xyplot(steps ~ interval | daytype, activity.complete.daytype,   
 type="l",   
 lwd=1,   
 xlab="Interval",   
 ylab="Mean Number of steps",   
 layout=c(1,2))



## It seems that the weekday activities starts earlier than the weekends and weekday activities starts around 5-6am and weekend activities starts around 8am.   
## Another observation is that from 10am to 5pm in the weekends have higher activity levels than the weekdays.