

## PROJECT RS 1

```
library(lattice)

## Warning: package 'lattice' was built under R version 3.3.3

activity_raw <- read.csv("C:/Users/hp-pc/Desktop/data/activity.csv",
stringsAsFactors=FALSE)
# Transform the date attribute to an actual date format
activity_raw$date <- as.POSIXct(activity_raw$date, format="%Y-%m-%d")

## Warning in strptime(x, format, tz = tz): unable to identify current
timezone 'H':
## please set environment variable 'TZ'

# Compute the weekdays from the date attribute
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"))

# Create the final data.frame
activity <- data.frame(date=activity_raw$date,
                      weekday=activity_raw$weekday,
                      daytype=activity_raw$daytype,
                      interval=activity_raw$interval,
                      steps=activity_raw$steps)

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

# Compute the total number of steps each day (NA values removed)
sum_data <- aggregate(activity$steps, by=list(activity$date), FUN=sum,
na.rm=TRUE)

# Rename the attributes
```

```

names(sum_data) <- c("date", "total")

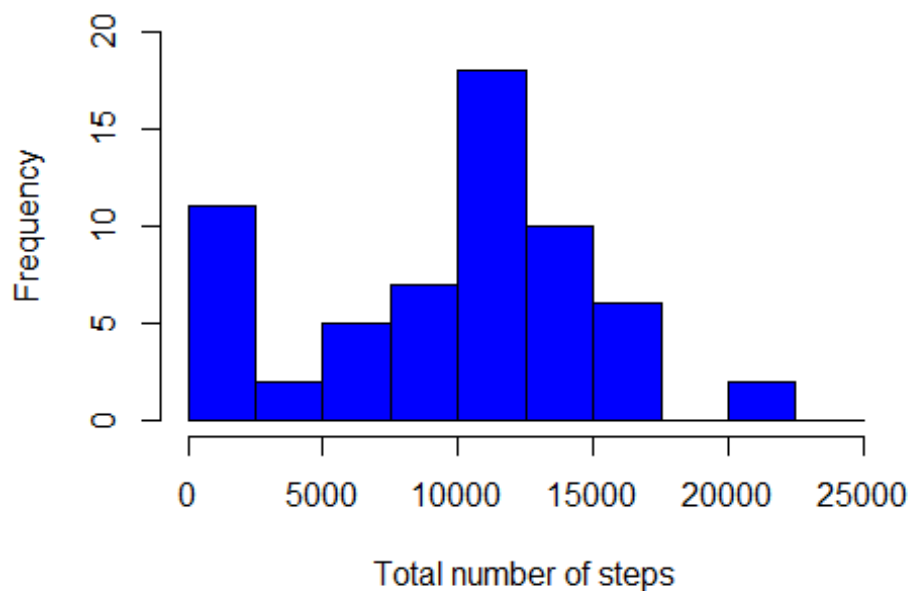
head(sum_data)

##      date total
## 1 2012-10-01     0
## 2 2012-10-02    126
## 3 2012-10-03 11352
## 4 2012-10-04 12116
## 5 2012-10-05 13294
## 6 2012-10-06 15420

# Compute the histogram of the total number of steps each day
hist(sum_data$total,
     breaks=seq(from=0, to=25000, by=2500),
     col="blue",
     xlab="Total number of steps",
     ylim=c(0, 20),
     main="Histogram of the total number of steps taken each day\n(NA
removed)")

```

**Histogram of the total number of steps taken each day  
(NA removed)**



```

mean(sum_data$total)

## [1] 9354.23

median(sum_data$total)

## [1] 10395

```

```

# Compute the means of steps accross all days for each interval
mean_data <- aggregate(activity$steps,
                        by=list(activity$interval),
                        FUN=mean,
                        na.rm=TRUE)

# Rename the attributes
names(mean_data) <- c("interval", "mean")

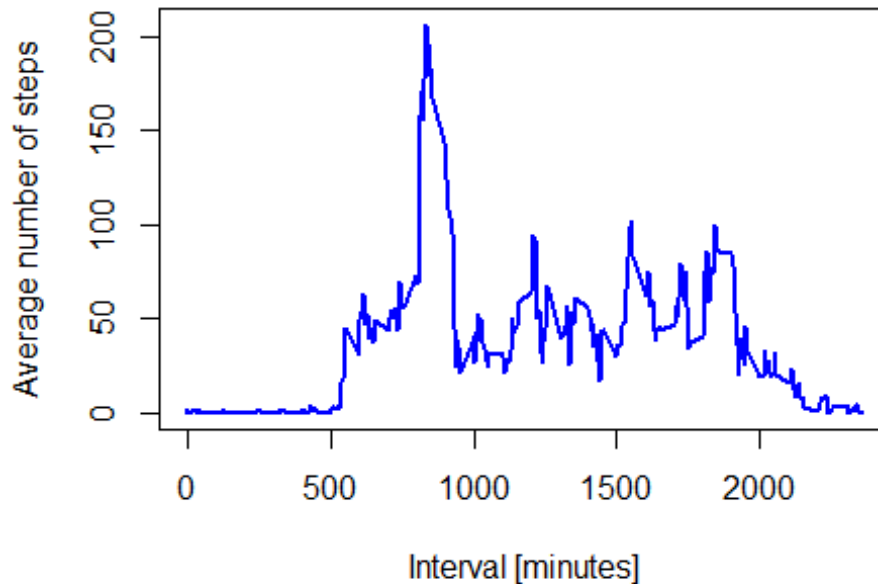
head(mean_data)

##   interval      mean
## 1         0 1.7169811
## 2         5 0.3396226
## 3        10 0.1320755
## 4        15 0.1509434
## 5        20 0.0754717
## 6        25 2.0943396

# Compute the time series plot
plot(mean_data$interval,
     mean_data$mean,
     type="l",
     col="blue",
     lwd=2,
     xlab="Interval [minutes]",
     ylab="Average number of steps",
     main="Time-series of the average number of steps per intervals\n(NA
removed)")

```

## Time-series of the average number of steps per inter (NA removed)



```
# We find the position of the maximum mean
max_pos <- which(mean_data$mean == max(mean_data$mean))

# We lookup the value of interval at this position
max_interval <- mean_data[max_pos, 1]

# We use the trick that a TRUE boolean value is equivalent to 1 and a FALSE
to 0.
NA_count <- sum(is.na(activity$steps))

# Find the NA positions
na_pos <- which(is.na(activity$steps))

# Create a vector of means
mean_vec <- rep(mean(activity$steps, na.rm=TRUE), times=length(na_pos))

# Replace the NAs by the means
activity[na_pos, "steps"] <- mean_vec

head(activity)
```

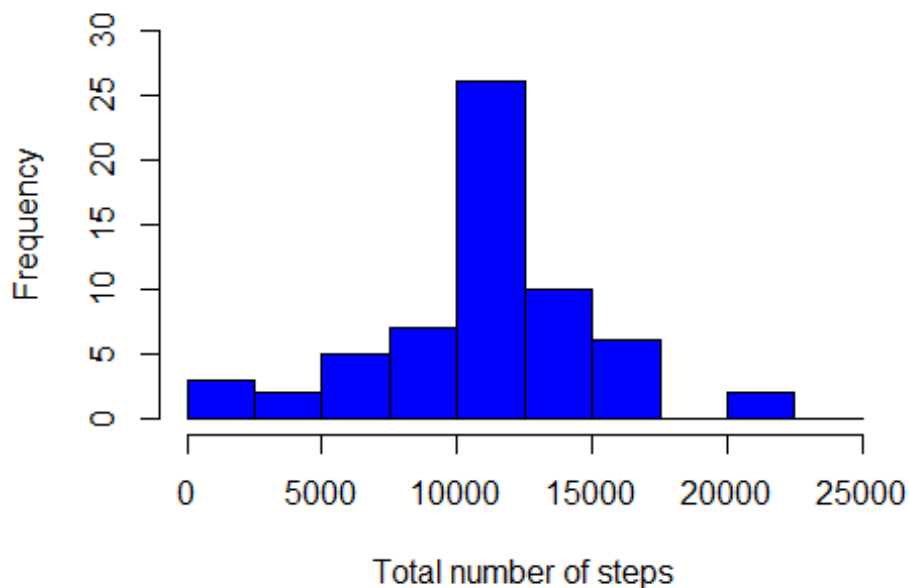
```
##           date weekday daytype interval  steps
## 1 2012-10-01  monday weekday         0 37.3826
## 2 2012-10-01  monday weekday         5 37.3826
## 3 2012-10-01  monday weekday        10 37.3826
## 4 2012-10-01  monday weekday        15 37.3826
## 5 2012-10-01  monday weekday        20 37.3826
## 6 2012-10-01  monday weekday        25 37.3826

# Compute the total number of steps each day (NA values removed)
sum_data <- aggregate(activity$steps, by=list(activity$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="blue",
     xlab="Total number of steps",
     ylim=c(0, 30),
     main="Histogram of the total number of steps taken each day\n(NA
replaced by mean value)")
```

**Histogram of the total number of steps taken each day  
(NA replaced by mean value)**



```
mean(sum_data$total)
## [1] 10766.19
median(sum_data$total)
```

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

```
# The new factor variable "daytype" was already in the activity data frame  
head(activity)
```

```
##      date weekday daytype interval  steps  
## 1 2012-10-01  monday weekday      0 37.3826  
## 2 2012-10-01  monday weekday      5 37.3826  
## 3 2012-10-01  monday weekday     10 37.3826  
## 4 2012-10-01  monday weekday     15 37.3826  
## 5 2012-10-01  monday weekday     20 37.3826  
## 6 2012-10-01  monday weekday     25 37.3826
```

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

```
mean_data <- aggregate(activity$steps,  
                        by=list(activity$daytype,  
                                activity$weekday, activity$interval), mean)
```

```
# Rename the attributes
```

```
names(mean_data) <- c("daytype", "weekday", "interval", "mean")
```

```
head(mean_data)
```

```
##  daytype weekday interval    mean  
## 1 weekday  friday      0 8.307244  
## 2 weekday  monday      0 9.418355  
## 3 weekend  saturday      0 4.672825  
## 4 weekend   sunday      0 4.672825  
## 5 weekday thursday      0 9.375844  
## 6 weekday  tuesday      0 0.000000
```

```
# Compute the time serie plot
```

```
xyplot(mean ~ interval | daytype, mean_data,  
       type="l",  
       lwd=1,  
       xlab="Interval",  
       ylab="Number of steps",  
       layout=c(1,2))
```

