RepData_PeerAssessment1

Alan Jenks

20/05/2019

Loading and preprocessing the data

#Code for reading in the dataset and/or processing the data

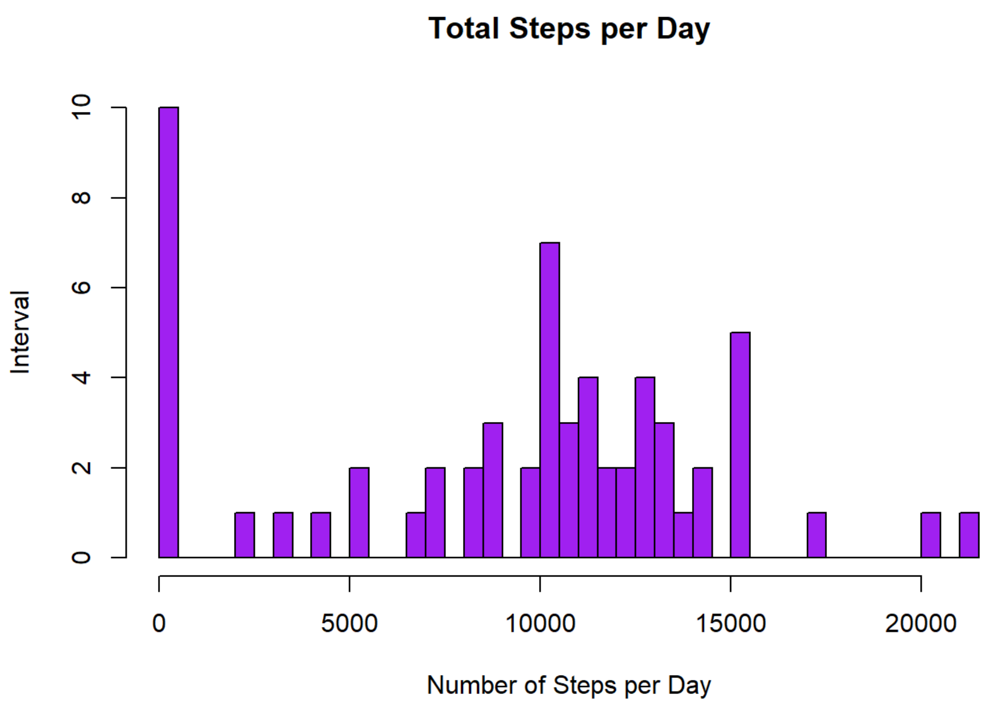
```
fileurl <- "https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip"
if (!file.exists('./repdata_data_activity.zip')){
  download.file(fileurl,'./repdata_data_activity.zip', mode = 'wb')
  unzip('./repdata_data_activity.zip', exdir = getwd())
}
activity <- read.csv('activity.csv')
str(activity)
```

```
## 'data.frame':    17568 obs. of  3 variables:
##  $ steps   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ date    : Factor w/ 61 levels "2012-10-01","2012-10-02",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ interval: int  0 5 10 15 20 25 30 35 40 45 ...
```

What is mean total number of steps taken per day?

#Histogram of the total number of steps taken each day

```
activity_steps <- tapply(activity$steps, activity$date, FUN=sum, na.rm=TRUE)

# Perform histogram of steps per day
hist(activity_steps,
    main="Total Steps per Day",
    xlab="Number of Steps per Day",
    ylab = "Interval",
    col="purple",
    breaks=50)
```



```
png("TotalStepsPerDay.png", width=480, height=480)
dev.off()
```

```
## png
##   2
```

#Mean and median number of steps taken each day

```
# Create mean and median of steps per day
stepsMean <- mean(activity_steps, na.rm=TRUE)
stepsMedian <- median(activity_steps, na.rm=TRUE)
# Output mean and median
stepsMean
```

```
## [1] 9354.23
```

```
stepsMedian
```

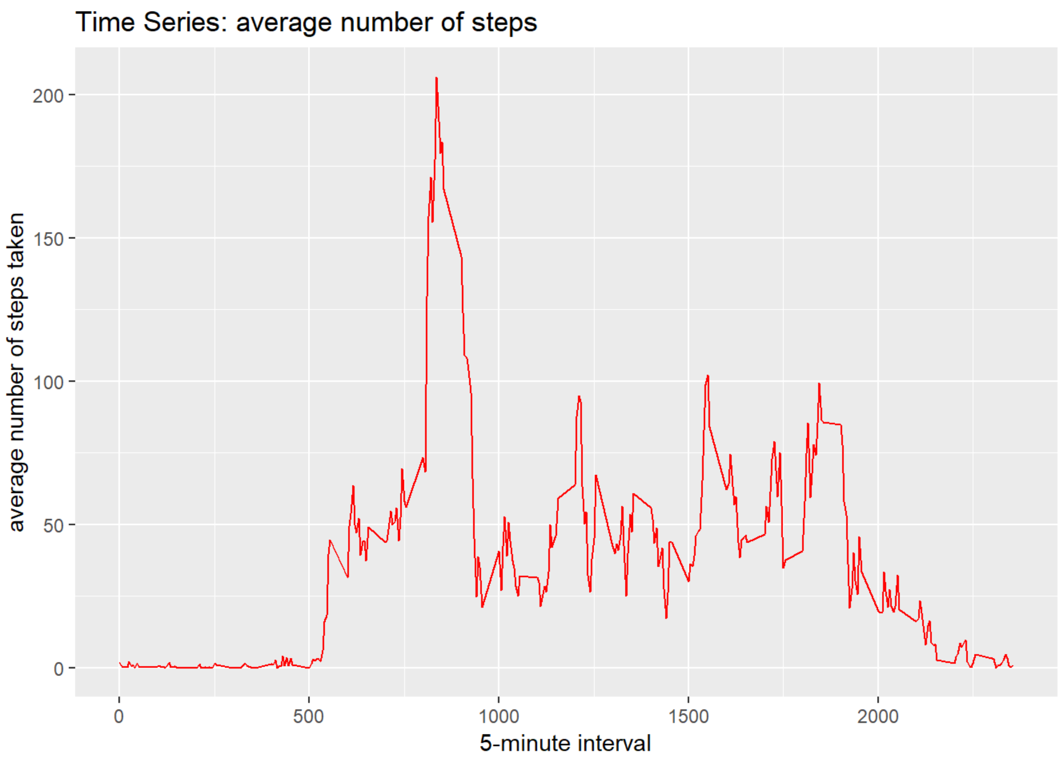
```
## [1] 10395
```

##Mean total number of steps per day is [b]9354[/b] ##Median number of steps is [b]10395[/b]

###What is the average daily activity pattern?

##Time series plot of the average number of steps taken

```
library(ggplot2)

# Create the means by intervals
averages <- aggregate(x=list(steps=activity$steps), by=list(interval=activity$interval),FUN=mean, na.rm=TRUE
)

ggplot(data=averages, aes(x=interval, y=steps),) +
  geom_line(color = "red") +
  ggtitle("Time Series: average number of steps") +
  xlab("5-minute interval") +
  ylab("average number of steps taken")
```



```
png("averagenumberofsteps.png", width=480, height=480)
dev.off()
```

```
## png
##   2
```

#The 5-minute interval that, on average, contains the maximum number of steps

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

```
##     interval    steps
## 104      835 206.1698
```

###Code to describe and show a strategy for imputing missing data

```
# copy of data frame
activity_missing <- activity

# add column for copleating index
activity_missing$CI <- "original"

# number of rows to check
l <- nrow(activity_missing)

# numbers of NAs
length(which(is.na(activity_missing$steps)))
```

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

```
for (i in 1:l) {
  if (is.na(activity_missing[i,1])) {
    activity_missing[i,1] <- averages[averages$interval == activity_missing[i,3],2]
    activity_missing[i,4] <- "completed"
  }
}

# numbers of NAs / completed (control)
length(which(is.na(activity_missing$steps)))
```

```
## [1] 0
```

```
length(which(activity_missing$CI=="completed"))
```

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

```
# Recreate the sums of steps per date
activity_steps2 <- tapply(activity_missing$steps, activity_missing$date, FUN=sum, na.rm=TRUE )

# Recreate the mean and median of steps per date
stepsMean2 <- mean(activity_steps2)
stepsMedian2 <- median(activity_steps2)

c(stepsMean2, stepsMean)
```

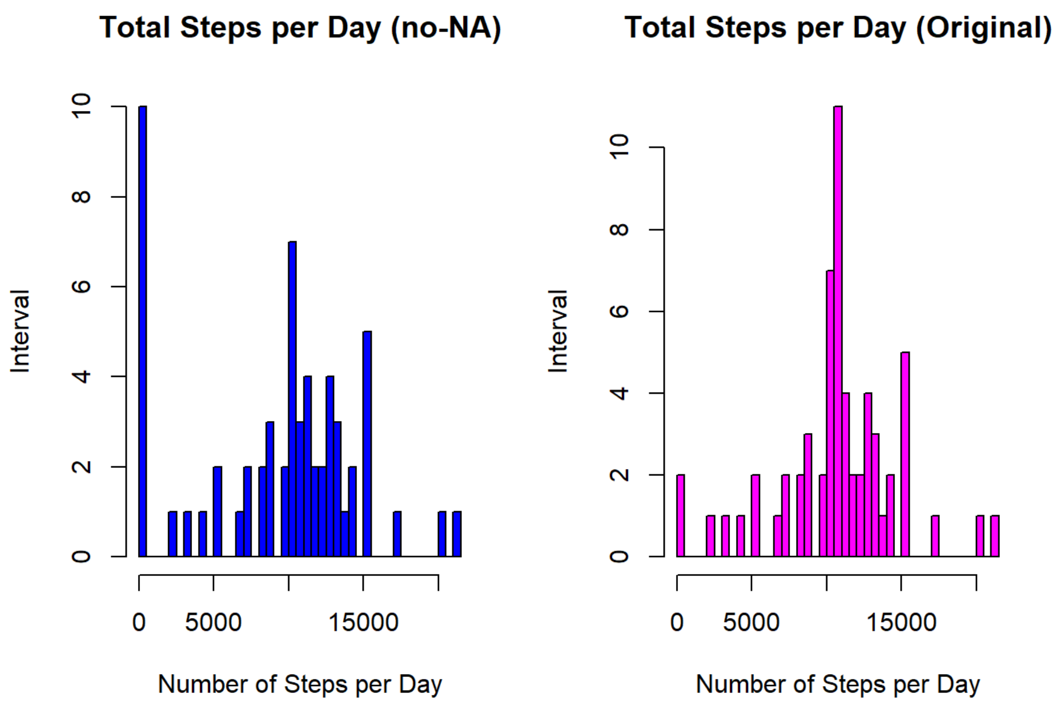
```
## [1] 10766.19  9354.23
```

```
c(stepsMedian2, stepsMedian)
```

```
## [1] 10766.19 10395.00
```

##Histogram of the total number of steps taken each day after missing values are imputed

```
# Plotting 2 plots in a grid
par(mfrow=c(1,2))
# Perform histogram of steps per day
hist(activity_steps,
    main = "Total Steps per Day (no-NA)",
    xlab = "Number of Steps per Day",
    ylab = "Interval",
    col="blue",
    breaks=50)
##Histogram with the orginal dataset
hist(activity_steps2,
    main="Total Steps per Day (Original)",
    xlab="Number of Steps per Day",
    ylab = "Interval",
    col="magenta",
    breaks=50)
```



```
png("noNavsOriginal.png", width=480, height=480)
dev.off()
```

```
## png
##   2
```

##Panel plot comparing the average number of steps taken per 5-minute interval across weekdays and weekends

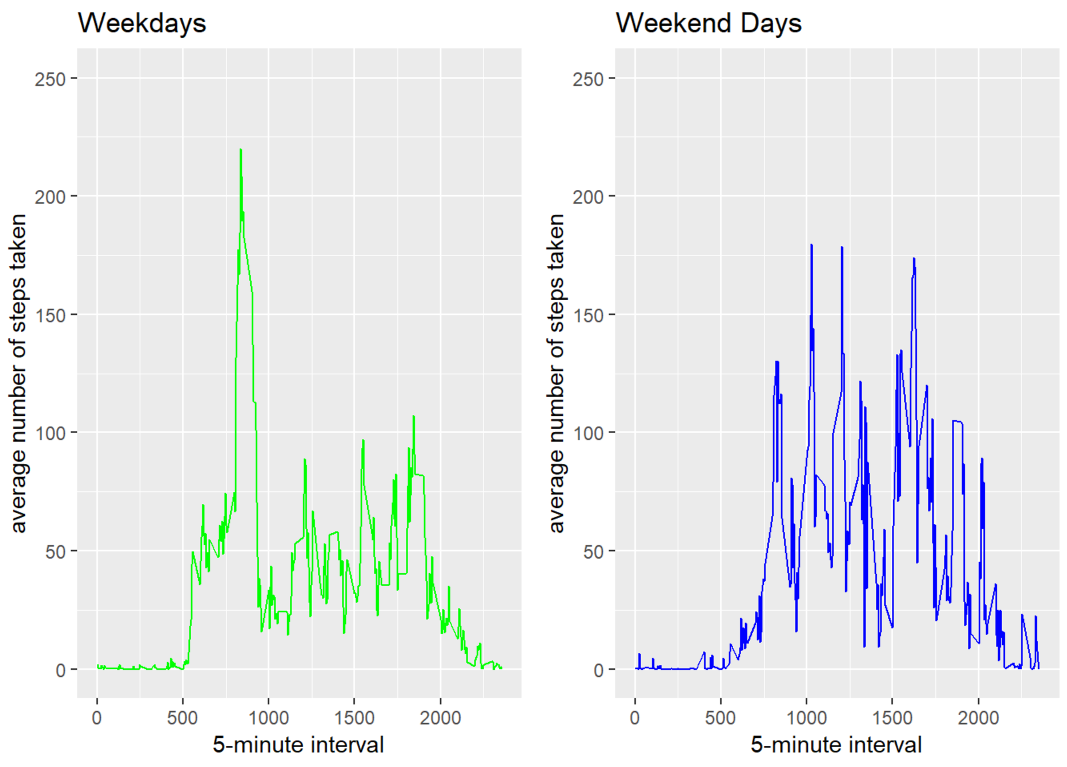
```
#create table with weekday and weekends
library(gridExtra)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
activity_missing[,2] <- as.Date(activity_missing[,2])
activity_missing$WD <- weekdays(activity_missing[,2])
activity_missing$WDG <- "week"               # default = "week"

# Filling in the WeekDayGroup in German
for (i in 1:l) {
  if (activity_missing[i,5] == "Saterday" | activity_missing[i,5] == "Sunday") {
    activity_missing[i,6] <- "weekend"
  }
}

activity_missing[,6] <- as.factor(activity_missing[,6])

activity_missingw <-subset(activity_missing,activity_missing[,6]=="week")
activity_missingwe <-subset(activity_missing,activity_missing[,6]=="weekend")

# Recreate the means by intervals
averagesW <- aggregate(steps ~ interval, activity_missingw, FUN=mean)
averagesWe <- aggregate(steps ~ interval, activity_missingwe, FUN=mean)

# prepare the plots
plot1 <- ggplot(data=averagesW, aes(x=interval, y=steps)) +
         geom_line(color = 'green') +
         ylim(0, 250) +
          ggtitle("Weekdays") +
         xlab("5-minute interval") +
         ylab("average number of steps taken")

plot2 <- ggplot(data=averagesWe, aes(x=interval, y=steps)) +
         geom_line(color = 'blue') +
         ylim(0, 250) +
         ggtitle("Weekend Days") +
         xlab("5-minute interval") +
         ylab("average number of steps taken")
grid.arrange(plot1, plot2, ncol=2)
```



```
png("weekdaysvsweekends.png", width=480, height=480)
dev.off()
```

```
## png
##   2
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.