PA1_template.Rmd

Hesham Abdelghany

March 7, 2016

Question 1:What is mean total number of steps taken per day?

The way I approached to solve this problem is by using dplyr package and to group_by the activity dataframe by days and then summarize the total number of steps using the summarize function so that the final total number of steps will be averaged over all the 5 mins interval in the day

Question 1 part 1: Calculate total steps per day

```
library(dplyr)
```

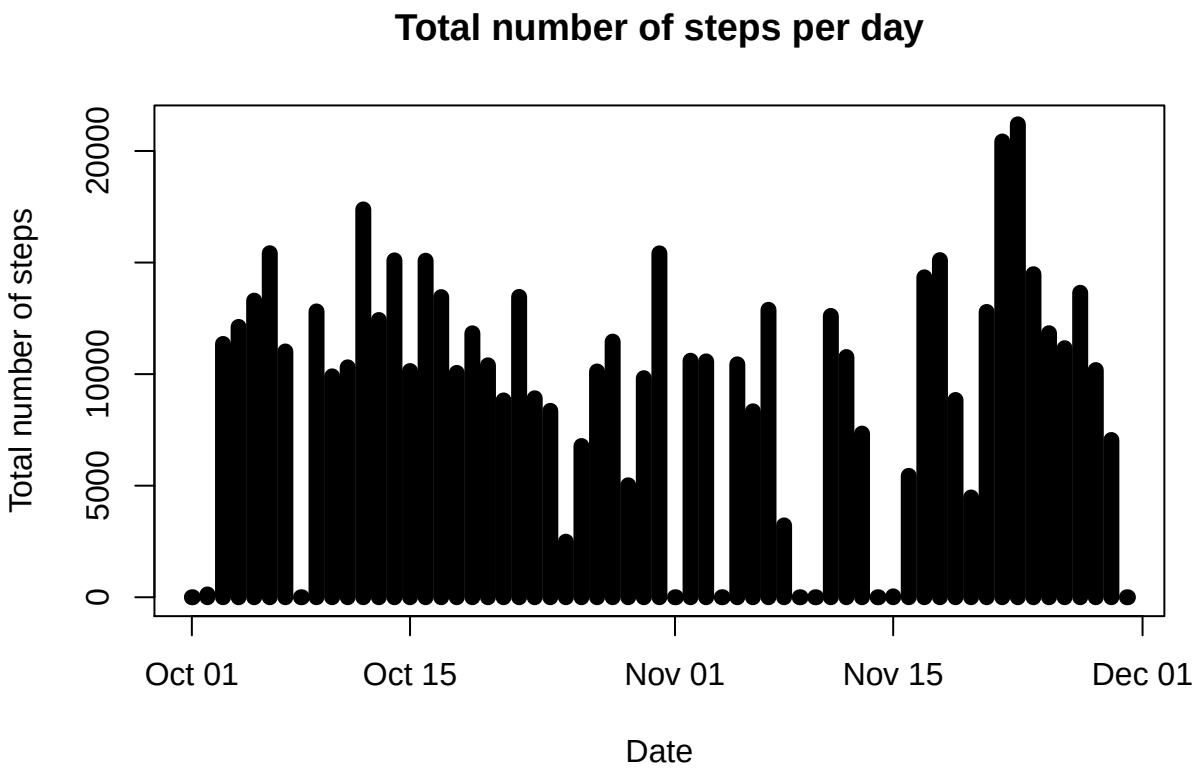
```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

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

```
activity <- read.csv("activity.csv")
activity$date <- as.Date(activity$date,"%Y-%m-%d")
activity_total_steps_day <- activity %>% group_by(date) %>% summarise(total_steps_day=sum(steps,na.rm=T
```

Question 1 part 2: plotting the total number of steps per day

Using the base plot function of type histogram, I am plotting the total number of steps per day as shown in the next code and followed by the graphs

```
with(activity_total_steps_day,plot(date,total_steps_day,type="h",lwd=8,xlab="Date",ylab="Total number o
```



Question 1 part 3: Reporting the mean and median of total steps per day

In this part, we are going to summarize the activity dataframe by the mean and the median as well as the total steps perday

```
activity_total_steps_day <- activity %>% group_by(date) %>% summarise(total_steps_day=sum(steps,na.rm=TR
```

The mean of the total number of steps per day is:

```
activity_total_steps_day$mean_steps_day
```

```
##  [1]        NaN  0.4375000 39.4166667 42.0694444 46.1597222 53.5416667
##  [7] 38.2465278        NaN 44.4826389 34.3750000 35.7777778 60.3541667
## [13] 43.1458333 52.4236111 35.2048611 52.3750000 46.7083333 34.9166667
## [19] 41.0729167 36.0937500 30.6284722 46.7361111 30.9652778 29.0104167
## [25]  8.6527778 23.5347222 35.1354167 39.7847222 17.4236111 34.0937500
## [31] 53.5208333        NaN 36.8055556 36.7048611        NaN 36.2465278
## [37] 28.9375000 44.7326389 11.1770833        NaN        NaN 43.7777778
## [43] 37.3784722 25.4722222        NaN  0.1423611 18.8923611 49.7881944
## [49] 52.4652778 30.6979167 15.5277778 44.3993056 70.9270833 73.5902778
## [55] 50.2708333 41.0902778 38.7569444 47.3819444 35.3576389 24.4687500
## [61]        NaN
```

The median of the total number of steps per day is:

```
activity_total_steps_day$median_steps_day
```

```
##  [1]    NA  63.0  61.0  56.5  66.0  67.0  52.5    NA  48.0  56.5  35.0
## [12]  46.0  45.5  60.5  54.0  64.0  61.5  52.5  74.0  49.0  48.0  52.0
## [23]  56.0  51.5  35.0  36.5  72.0  61.0  54.5  40.0  83.5    NA  55.5
## [34]  59.0    NA  66.0  52.0  58.0  42.5    NA    NA  55.0  42.0  57.0
## [45]    NA  20.5  43.0  65.5  80.0  34.0  58.0  55.0  65.0 113.0  65.5
## [56]  84.0  53.0  57.0  70.0  44.5    NA
```

Question 2: What is the average daily activity pattern?

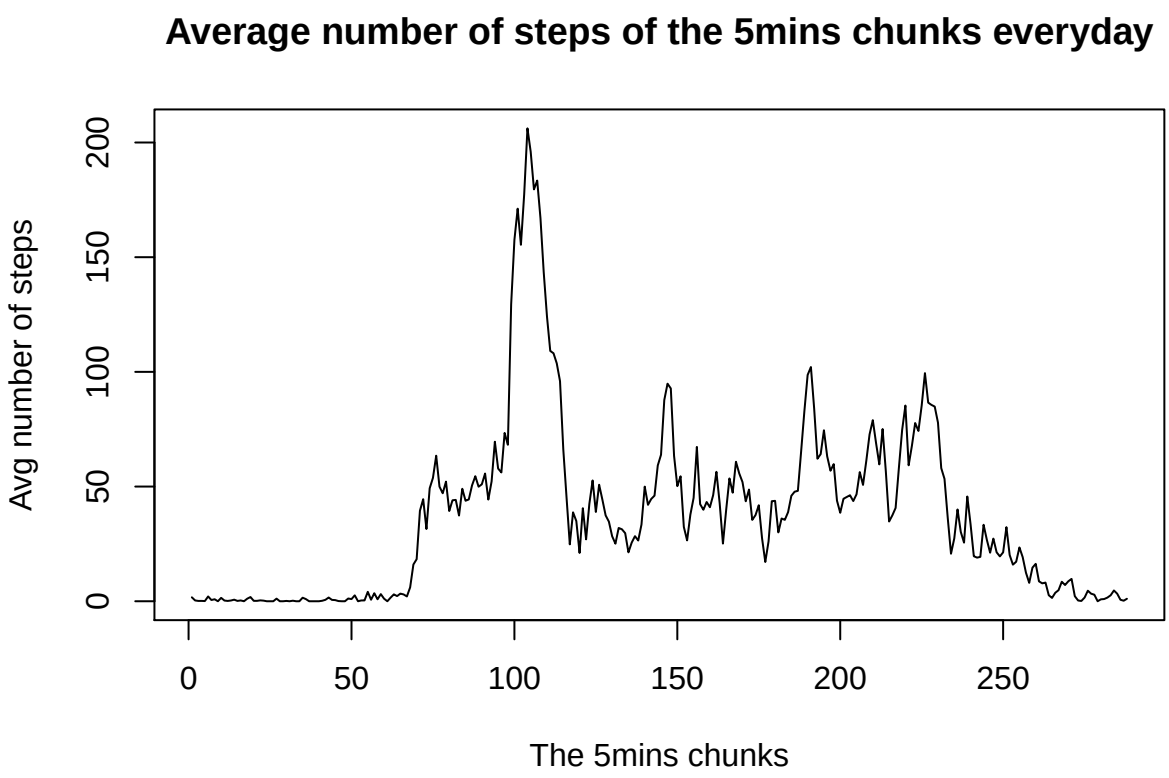
Here we want to see the daily activity for each 5 mins interval.So (24 hours * 60 minutes) / 5 minutes interval = 288 groups of 5 mins interval throughout the 24 hours of the day. So what I need to do here is to figure out the total number of steps along these 288 chunks everyday and see when is the most active time (5mins interval in the day)

So first we are going to have these 288 chunks of 5 mins intervals multiplied by the total number of days. First, we are going to mutate the original activity dataframe with the 288 chunks*61days Second, Group the activity dataframe by the 5mins interval chunks then summarize by the avg number of steps fo each 5mins chunk Third, plot the avg steps per each 5mins chunk Fourth, report which 5mins interval has the max number of steps

```
mins_intervals <- rep(c(1:288),nrow(activity_total_steps_day))
activity <- activity %>% mutate(min5_chunks=mins_intervals,which_day=weekdays(date))
activity_mins_interval <- activity %>% group_by(min5_chunks) %>% summarize(avg_steps_min5_interval=mean
```

In the following plot, you will see that around the 100th 5min chunk in everyday, will be the most active time. Calculating what does that mean (100/288)*24hours of the day ~ between 8am and 9am in the morning which makes sense. and in the very first and very last chunks, the data is almost zero where the day is just starting or ending repectively

```
with(activity_mins_interval,plot(min5_chunks,avg_steps_min5_interval,type="l",xlab="The 5mins chunks",y
```



Which 5mins interval between the 288 chunks of the day has the max activity?

```
which.max(activity_mins_interval$avg_steps_min5_interval)
```

```
## [1] 104
```

Question 3: Imputting missing data

My methodology here toimput missing data is to find the mean of the missing 5mins interval across the same day and put that in the missing place

Here, I am reporting the number of missing values in the steps variable of the activity data frame:

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

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

Here is the methodology for imputting the data:

```
for (i in 1:nrow(activity)) {
  if (is.na(activity[i,]$steps)) {
    this_day <- activity[i,]$which_day
    this_chunk <- activity[i,]$min5_chunks
    this_mean <- mean(activity[activity$min5_chunks==this_chunk&activity$which_day==this_day,]$steps,na
```

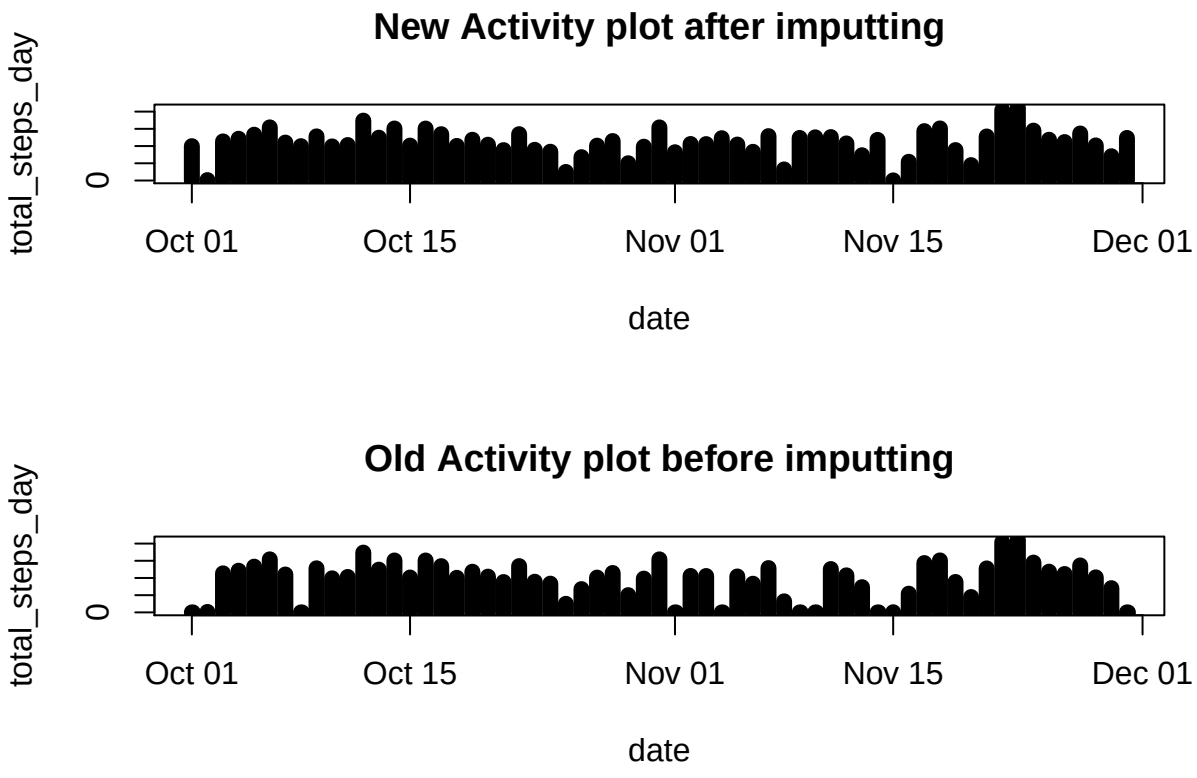
```
    activity[i,]$steps <- this_mean
  }
}
```

Comparingactivity dataframes before and after the imputting:

```
activity_total_steps_day <- activity %>% group_by(date) %>% summarise(total_steps_day=sum(steps,na.rm=TR

old_activity <- read.csv("activity.csv")
old_activity$date <- as.Date(old_activity$date,"%Y-%m-%d")

old_activity_total_steps_day <- old_activity %>% group_by(date) %>% summarise(total_steps_day=sum(steps
```

Here is the plot

```
par(mfrow=c(2,1))
with(activity_total_steps_day,plot(date,total_steps_day,type="h",lwd=8,main="New Activity plot after imp
with(old_activity_total_steps_day,plot(date,total_steps_day,type="h",lwd=8,main="Old Activity plot befo
```




```
par(mfrow=c(1,1))
```

Reporting the mean and median for the dataframe before imputting:

```
old_activity_total_steps_day <- old_activity %>% group_by(date) %>% summarise(total_steps_day=sum(steps

old_activity_total_steps_day$mean_steps_day
```

```
##  [1]        NaN  0.4375000 39.4166667 42.0694444 46.1597222 53.5416667
##  [7] 38.2465278        NaN 44.4826389 34.3750000 35.7777778 60.3541667
## [13] 43.1458333 52.4236111 35.2048611 52.3750000 46.7083333 34.9166667
## [19] 41.0729167 36.0937500 30.6284722 46.7361111 30.9652778 29.0104167
## [25]  8.6527778 23.5347222 35.1354167 39.7847222 17.4236111 34.0937500
## [31] 53.5208333        NaN 36.8055556 36.7048611        NaN 36.2465278
## [37] 28.9375000 44.7326389 11.1770833        NaN        NaN 43.7777778
## [43] 37.3784722 25.4722222        NaN  0.1423611 18.8923611 49.7881944
## [49] 52.4652778 30.6979167 15.5277778 44.3993056 70.9270833 73.5902778
## [55] 50.2708333 41.0902778 38.7569444 47.3819444 35.3576389 24.4687500
## [61]        NaN
```

```
old_activity_total_steps_day$median_steps_day
```

```
##  [1]    NA  63.0  61.0  56.5  66.0  67.0  52.5    NA  48.0  56.5  35.0
## [12]  46.0  45.5  60.5  54.0  64.0  61.5  52.5  74.0  49.0  48.0  52.0
## [23]  56.0  51.5  35.0  36.5  72.0  61.0  54.5  40.0  83.5    NA  55.5
## [34]  59.0    NA  66.0  52.0  58.0  42.5    NA    NA  55.0  42.0  57.0
## [45]    NA  20.5  43.0  65.5  80.0  34.0  58.0  55.0  65.0 113.0  65.5
## [56]  84.0  53.0  57.0  70.0  44.5    NA
```

```
AfterImput_activity_total_steps_day <- activity %>% group_by(date) %>% summarise(total_steps_day=sum(st

AfterImput_activity_total_steps_day$mean_steps_day
```

```
##  [1] 34.6349206  0.4375000 39.4166667 42.0694444 46.1597222 53.5416667
##  [7] 38.2465278 34.6349206 44.4826389 34.3750000 35.7777778 60.3541667
## [13] 43.1458333 52.4236111 35.2048611 52.3750000 46.7083333 34.9166667
## [19] 41.0729167 36.0937500 30.6284722 46.7361111 30.9652778 29.0104167
## [25]  8.6527778 23.5347222 35.1354167 39.7847222 17.4236111 34.0937500
## [31] 53.5208333 28.5164931 36.8055556 36.7048611 42.6309524 36.2465278
## [37] 28.9375000 44.7326389 11.1770833 42.9156746 43.5257937 43.7777778
## [43] 37.3784722 25.4722222 40.9401042  0.1423611 18.8923611 49.7881944
## [49] 52.4652778 30.6979167 15.5277778 44.3993056 70.9270833 73.5902778
## [55] 50.2708333 41.0902778 38.7569444 47.3819444 35.3576389 24.4687500
## [61] 42.9156746
```

```
AfterImput_activity_total_steps_day$median_steps_day
```

```
##  [1]  24.71429  63.00000  61.00000  56.50000  66.00000  67.00000  52.50000
##  [8]  24.71429  48.00000  56.50000  35.00000  46.00000  45.50000  60.50000
## [15]  54.00000  64.00000  61.50000  52.50000  74.00000  49.00000  48.00000
## [22]  52.00000  56.00000  51.50000  35.00000  36.50000  72.00000  61.00000
## [29]  54.50000  40.00000  83.50000  15.12500  55.50000  59.00000  45.00000
## [36]  66.00000  52.00000  58.00000  42.50000  30.07143  39.28571  55.00000
## [43]  42.00000  57.00000  25.68750  20.50000  43.00000  65.50000  80.00000
## [50]  34.00000  58.00000  55.00000  65.00000 113.00000  65.50000  84.00000
## [57]  53.00000  57.00000  70.00000  44.50000  30.07143
```

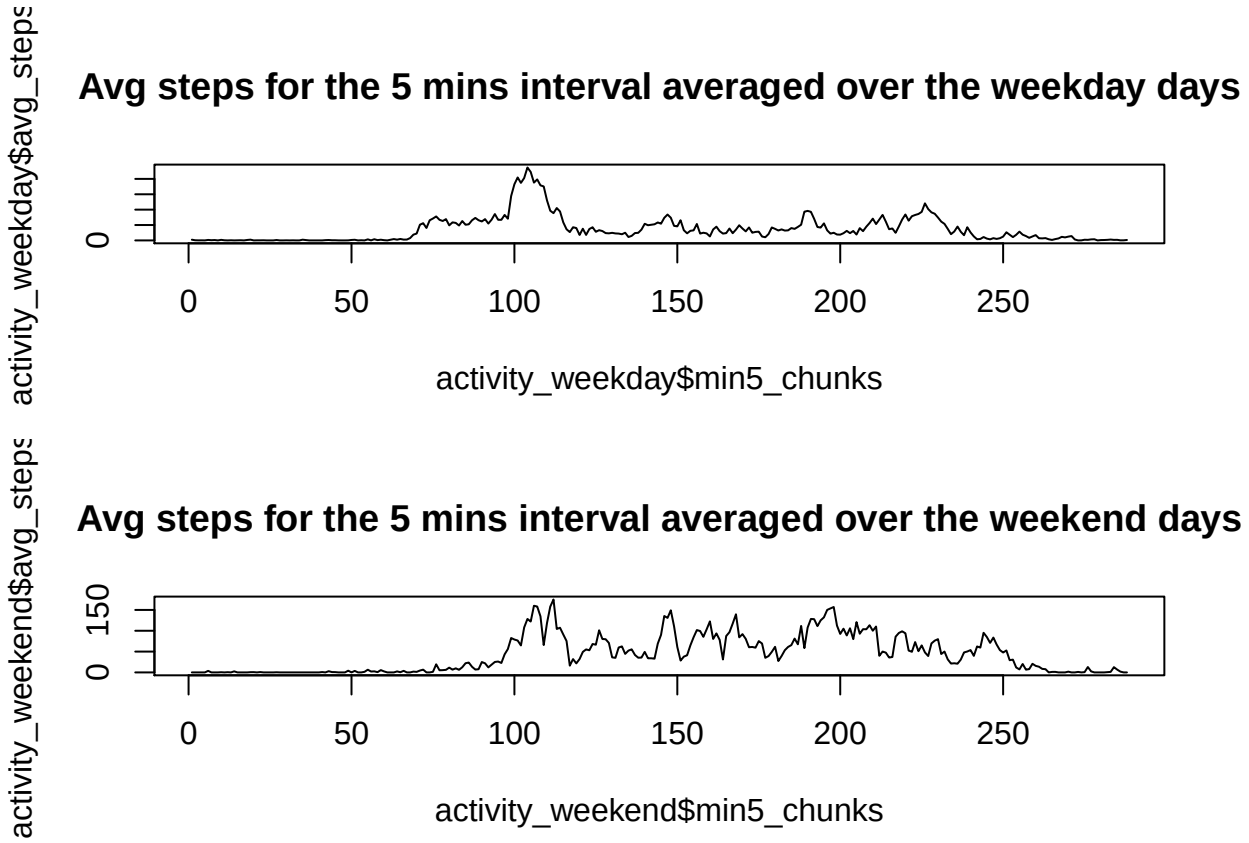
Comparing the steps during weekdays and weekends

Here I am going to split the activity dataframe into weekends data and weekdays data and plot the comparison and see how it changes.

```
activity <- activity %>% mutate(weekday_or_end=ifelse(which_day=="Saturday"|which_day=="Sunday",c("week
activity$weekday_or_end <- as.factor(activity$weekday_or_end)
activity_subset <- activity %>% group_by(weekday_or_end,min5_chunks) %>% summarize(avg_steps=mean(steps

activity_weekday <- subset(activity_subset,weekday_or_end=="weekday")
activity_weekend <- subset(activity_subset,weekday_or_end=="weekend")
```

As will be shown in the following plot, that the activity during weekends are more than the activity during weekdays:

```
par(mfrow=c(2,1))
plot(activity_weekday$min5_chunks,activity_weekday$avg_steps,type="l",main="Avg steps for the 5 mins int
plot(activity_weekend$min5_chunks,activity_weekend$avg_steps,type="l",main="Avg steps for the 5 mins int
```




```
par(mfrow=c(1,1))
```

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this: