Reproducible Research - Course Project 1

glenzlaoshi

12/14/2020

Introduction

Welcome. So we are given some data representing the number of steps of an anonymous subject obtained during the months of October and November, 2012. Information provided tells us that the dataset consists of the following variables:

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

Let's see what we can make of this data.

Loading and Preprocessing

First, we will read the data from the 'activity.csv' file in the directory, and save it as a dataframe using the 'read.csv' function. We can then get some basic information about this dataframe and it's organization and contents.

```
library(dplyr)
library(ggplot2)
data<-(read.csv("activity.csv"))
head(data)
```

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

```
str(data)
```

```
## '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 ...
```

Exploratory Analysis

As can be seen, this dataframe consists of 3 variables (steps, date, interval) and a total of 17568 observations.

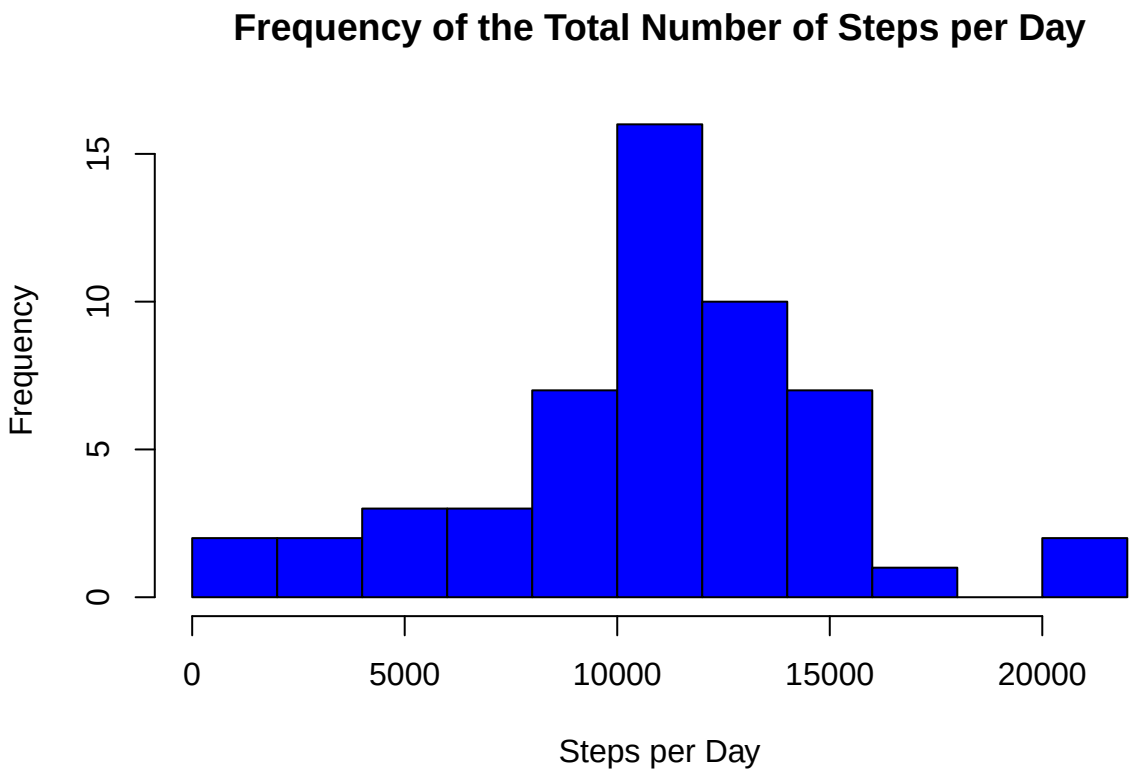
Let's start the exploratory analysis by cleaning up the data a bit, by using the 'dplyr' package to combine (sum) the number of steps for the same day and get rid of multiples of the same date.

```
data$date <- as.Date(data$date,format="%Y-%m-%d")

complete_data <- data[complete.cases(data),]
step_data <- complete_data %>%
    group_by(date) %>%
    summarize(steps = sum(steps))
```

Now let's create a histogram of the total number of steps taken in each day.

```
hist(step_data$steps,breaks=10,col='blue',main="Frequency of the Total Number of Steps per Day",xlab="S
```



From this graph, it looks like you could estimate that both the mean and median are a little more than 10,000 steps/day; let's check this out.

```
mean(step_data$steps)
```

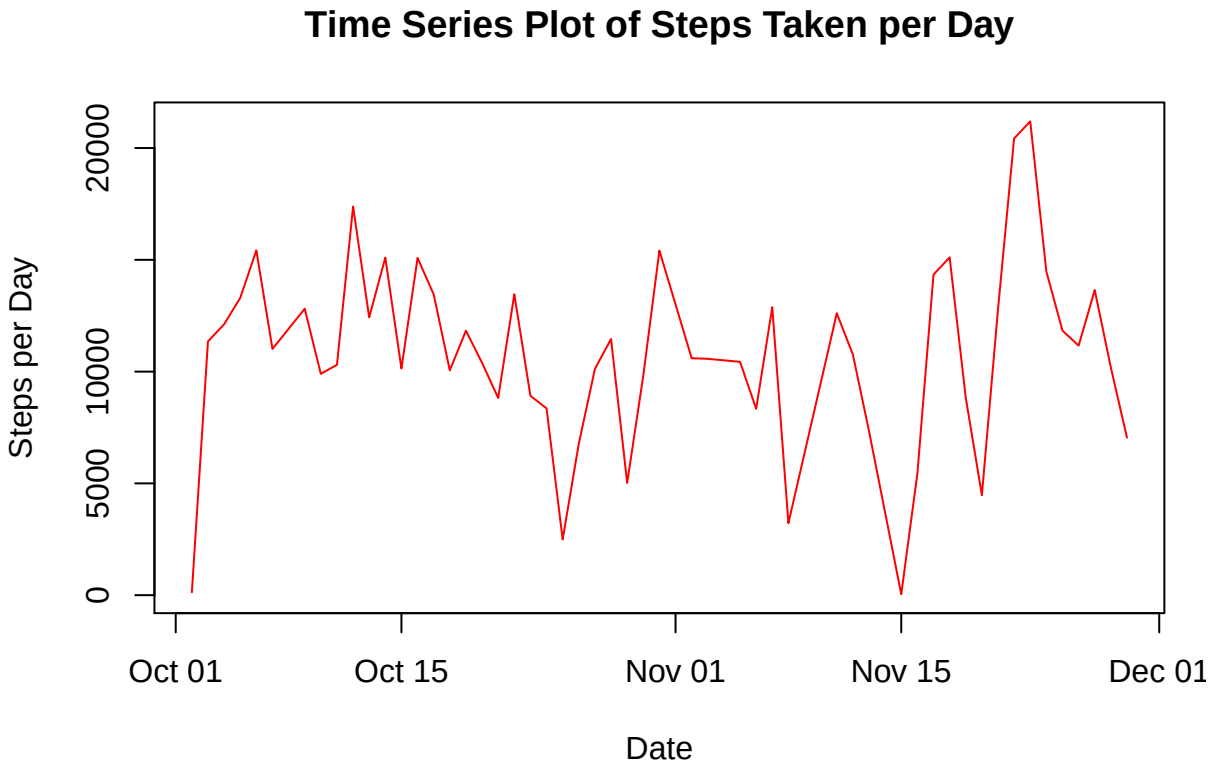
```
## [1] 10766.19
```

```
median(step_data$steps)
```

```
## [1] 10765
```

What if we look at the number of steps per day?

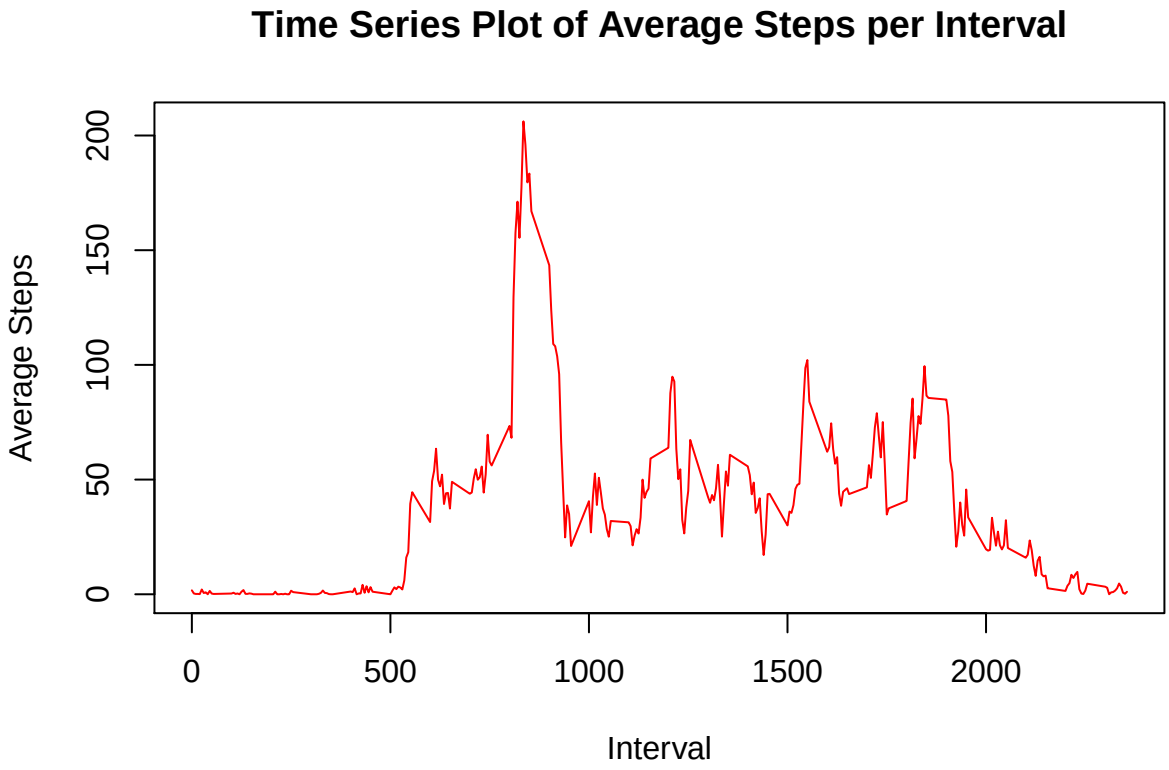
```
plot(step_data$steps~step_data$date,type='l',main="Time Series Plot of Steps Taken per Day", ylab="Steps
```



I think there are a number of possible questions we could ask regarding this data (i.e. general trend over time), however, it is very dependent upon the data subject, and may not meaningfully be applied to others.

What if we want to look at the average number of steps during the various time intervals? First we would have to re-arrange our data and sort by interval, then we could graph it to gain some possible insight.

```
interval_data <- complete_data %>%
    group_by(interval) %>%
    summarize(steps = mean(steps))
plot(interval_data$steps~interval_data$interval,type='l',main="Time Series Plot of Average Steps per In
```



We can see from the graph that the maximum number of steps, on average, occurs at some interval between 750 and 1000. We could find out when this actually is a number of ways. For example we could find out what the maximum number of steps, on average, is; and then use that to find the interval where it occurs:

```
max(interval_data$steps)
```

```
## [1] 206.1698
```

```
interval_data$interval[which(interval_data$steps == max(interval_data$steps))]
```

```
## [1] 835
```

Or, probably easier, we could use the built-in 'which.max' directly:

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

```
## # A tibble: 1 x 2
##   interval steps
##      <int> <dbl>
## 1      835  206.
```

Both of these result in the 5-second interval at 835 being when the maximum number of steps, on average, is reached with respect to the given data. I suppose if you think about it, this seems reasonable, being around 8:30 am, a prime moving about time.

Missing Values

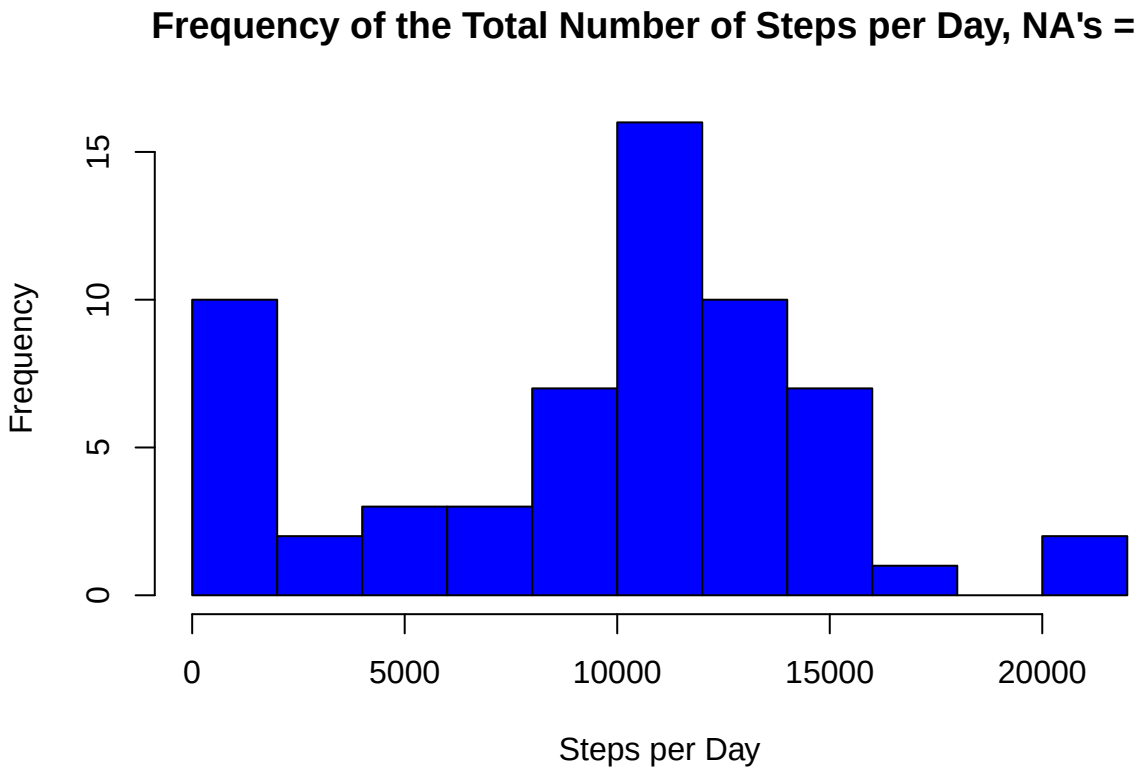
At the very beginning, I used the 'complete.cases' function to remove all of the rows containing NA's from the data. But what if we did not want to remove all of that data? How many entries contain NA in the original data? Could we replace the NA's with some other values?

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

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

From the above code, we can see that there are 2304 NA's in the original data. What if we replace these with zeros?

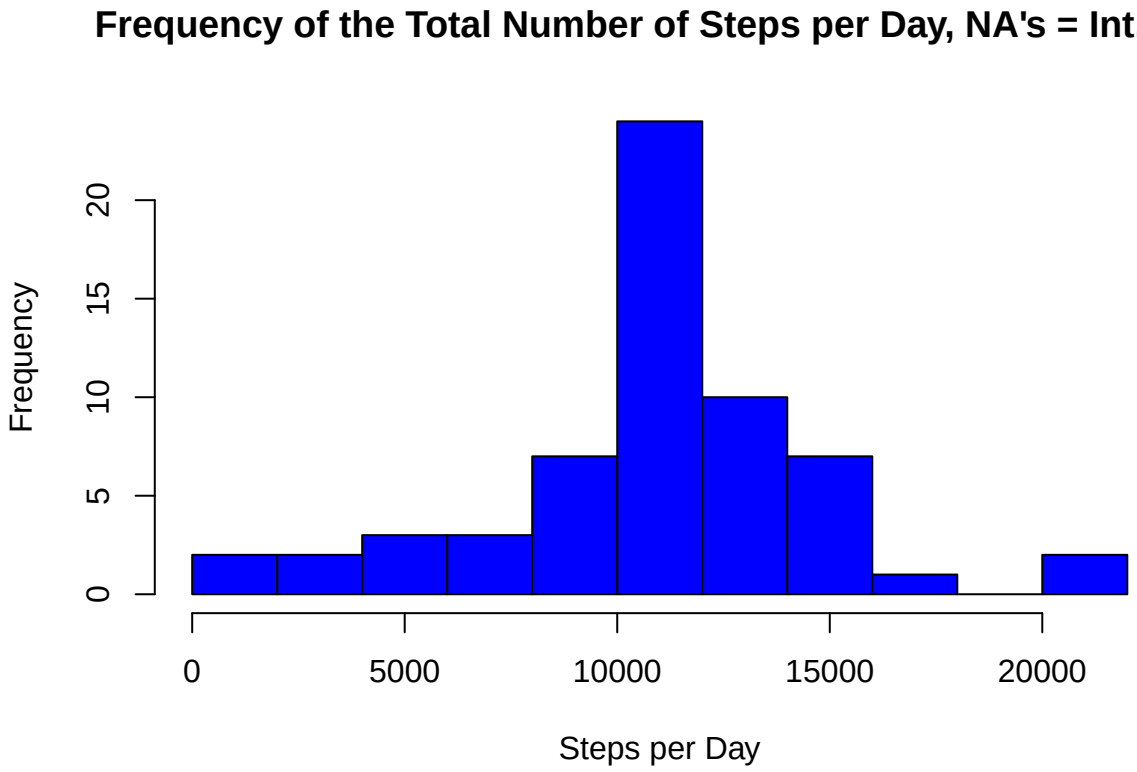
```
completed_data <- data
na_values <- is.na(completed_data$steps)
completed_data$steps[na_values] <- 0
completed_step_data <- completed_data %>%
    group_by(date) %>%
    summarize(steps = sum(steps))
hist(completed_step_data$steps,breaks=10,col='blue',main="Frequency of the Total Number of Steps per Da
```



This looks similar to the original histogram, except for the large column corresponding to zero steps. We replaced 2304 NA's with zeros, but this corresponds to 13.1147541% of the data, and obviously creates an outsized influence on the data. In addition, this also means that there were many days within the dataset that now have zero total steps, which is highly unlikely within the context of how the data was gathered. So, what might a better replacement option be?

One relatively easy replacement fix might be to replace the NA's with the overall average number of steps per interval. We can find this by using the average (mean) number of steps in a day we found before (1.0766189×10^4), and dividing this by the total number of intervals in a day (288), which gives an average number of steps per interval of 37.3825996.

```
interval_average <- mean(step_data$steps)/288
completed_data <- data
na_values <- is.na(completed_data$steps)
completed_data$steps[na_values] <- interval_average
completed_step_data <- completed_data %>%
    group_by(date) %>%
    summarize(steps = sum(steps))
hist(completed_step_data$steps,breaks=10,col='blue',main="Frequency of the Total Number of Steps per Da
```




```
mean(completed_step_data$steps)
```

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

```
median(completed_step_data$steps)
```

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

In reality, this is not much better than before, as now we have just increased the size of the one bin. Most likely, this is because there are a large number of days within the data that have only NA's, which means we now gave those days a value of steps per day corresponding to the mean. This does not alter the overall average (seen in the code above), but does not offer any additional insight either. It would be much more useful to ignore the NA's as originally done, or find some other way to replace them that might be more meaningful.

Relaxing Weekends?

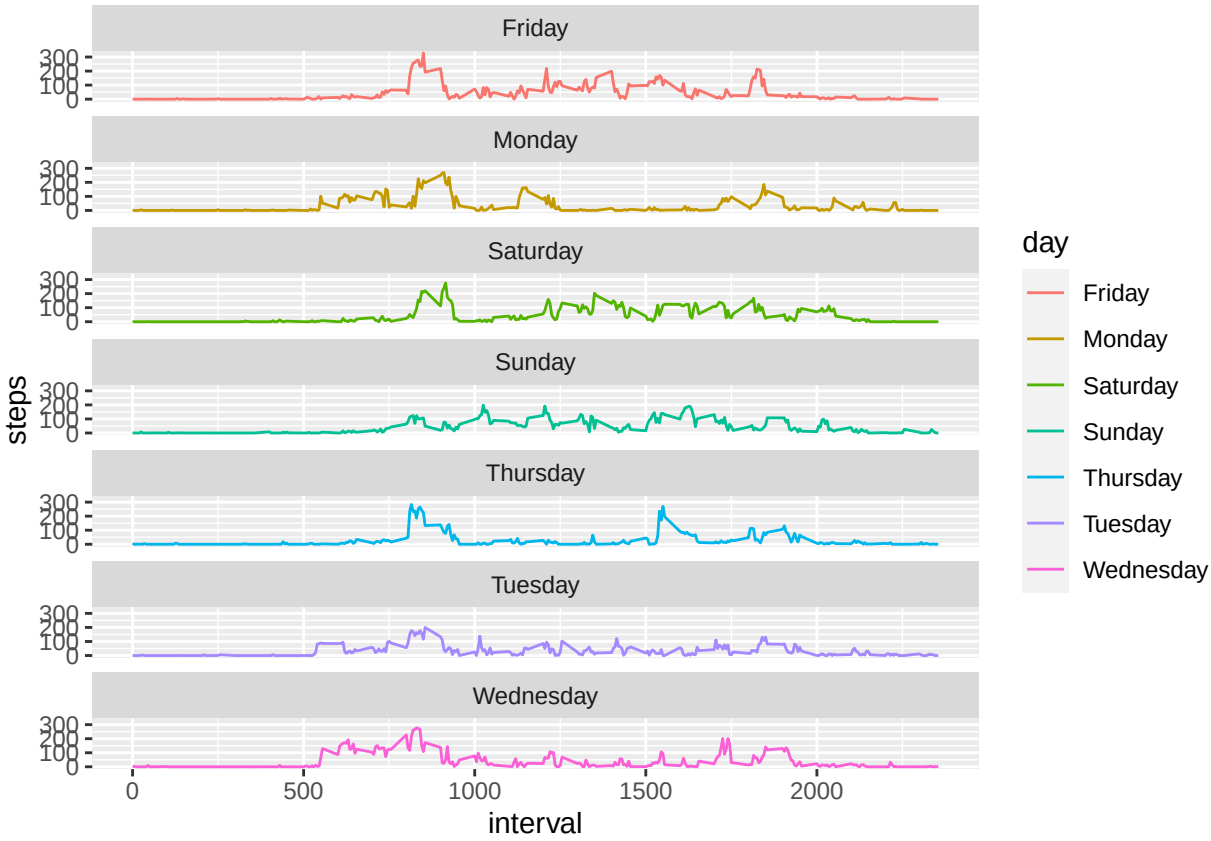
When looking at this data, a common question someone might have is whether or not there is a difference in steps taken during weekends versus weekdays? There are several ways we could visualize this, but let's just compare two plots, similar to what we did before, but split the data out by weekend and weekday. Let's assume that the weekend consists of Saturday and Sunday.

```
complete_day_data <- complete_data %>%
    mutate(day = weekdays(complete_data$date))
head(complete_day_data)
```

```
##   steps       date interval     day
## 1     0 2012-10-02        0 Tuesday
## 2     0 2012-10-02        5 Tuesday
## 3     0 2012-10-02       10 Tuesday
## 4     0 2012-10-02       15 Tuesday
## 5     0 2012-10-02       20 Tuesday
```

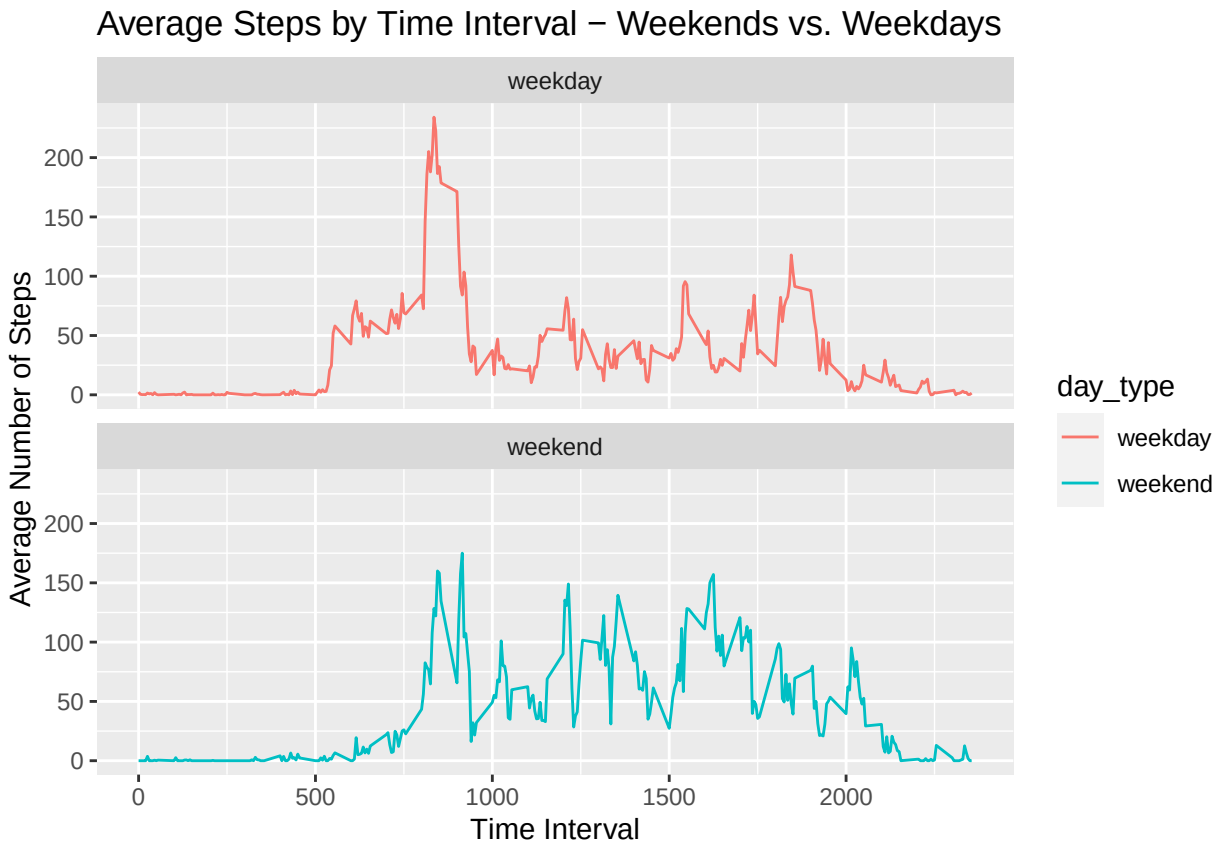
```
## 6     0 2012-10-02       25 Tuesday
```

```
interval_data <- complete_day_data %>%
    group_by(interval,day) %>%
    summarize(steps = mean(steps))
ggplot(interval_data, aes(x=interval, y=steps, color=day))+
    geom_line()+
    facet_wrap(~day,ncol=1,nrow=7)
```



The above graph shows the time series for all days of the week. I am sure we can ask some really great questions about what might be shown on different days, but for now let's get back to comparing weekends to weekdays.

```
complete_day_type_data <- mutate(complete_day_data, day_type = ifelse(complete_day_data$day!="Saturday"
interval_data <- complete_day_type_data %>%
    group_by(interval,day_type) %>%
    summarize(steps = mean(steps))
ggplot(interval_data, aes(x=interval, y=steps, color=day_type))+
    geom_line()+
    labs(title="Average Steps by Time Interval - Weekends vs. Weekdays",x="Time Interval",y="Average Nu
    facet_wrap(~day_type,ncol=1,nrow=2)
```



Alright! We can now try to attach some rationalizations to this visualization. The subject appears to start moving around later on the weekends (sleeping in a bit?), but is also moving more during the day than on weekdays (not sitting down working all day?) All this seems to make sense, at least initially and based upon what we know of the data provided.

I trust you have enjoyed this fascinating foray into preliminary data analysis, and I hope this R-markdown document has allowed you to follow along as necessary.