Mod5Wk2_Peer

Gregg Powell

11/4/2020

Module 5 Week 2 Peer Reviewed Assignment

Assignment Instructions:

##Load the data

Process/transform the data (if necessary) into a format suitable for your analysis What is mean total number of steps taken per day? For this part of the assignment, you can ignore the missing values in the dataset.

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

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

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

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

```
data <- fread("C:/_TEMP/_R_WORK_TEMP/Johns_Hopkins_Data_Science/Module 5-Reproducable_Research/Week2/peer_assignment/activity.csv")
```

##What is mean total number of steps taken per day?

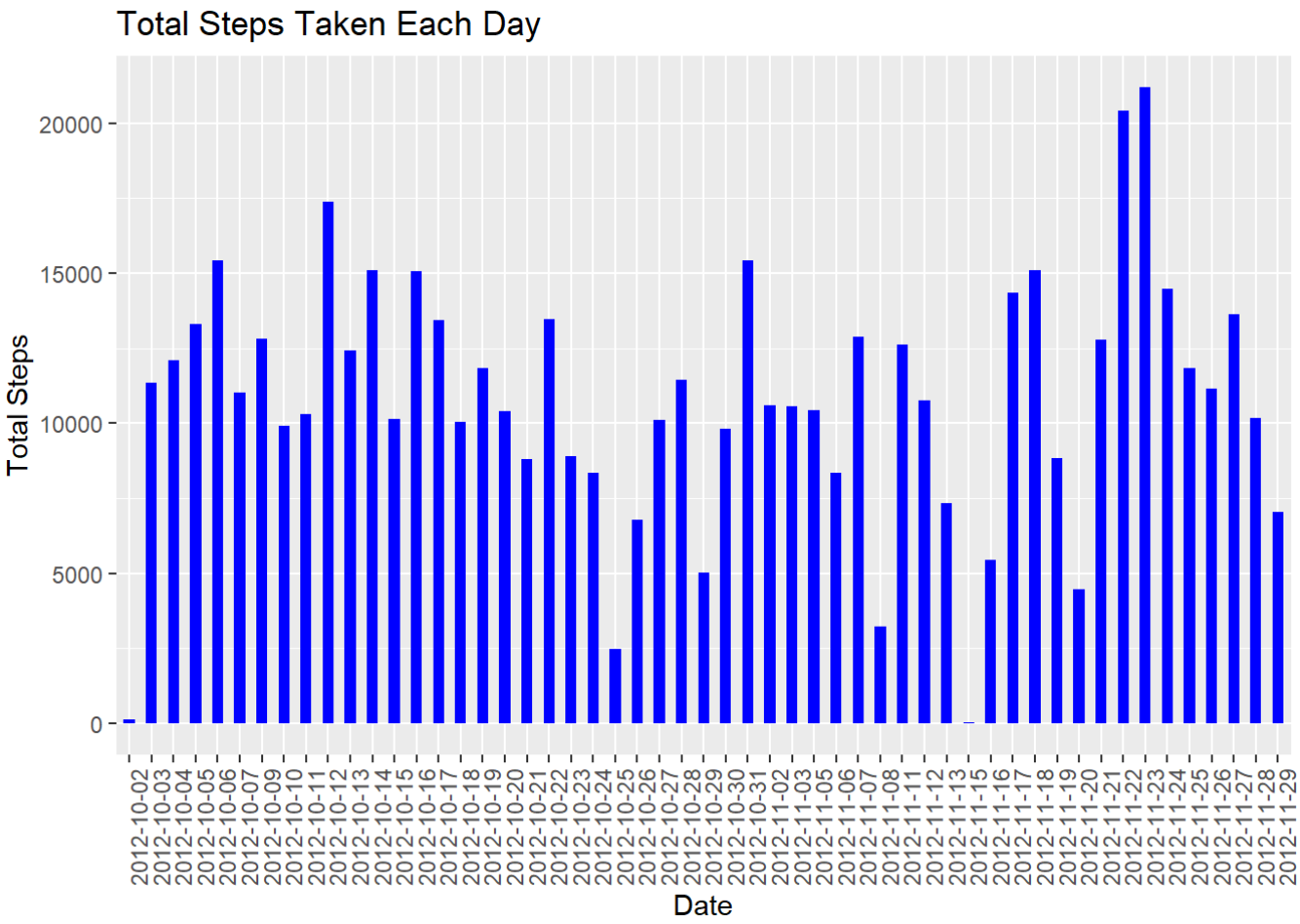
Calculate the total number of steps taken per day

```
#remove all rows with NA or with "" (nothing)
data_no_NA <- data[!(is.na(data$steps) | data$steps==""), ]
#calculate the number of steps and print
mean_steps <- mean(data_no_NA$steps)
mean_steps
```

```
## [1] 37.3826
```

Make a histogram of the total number of steps taken each day

```
#Plots a histogram where the y-axis is the total steps and the x-axis indicates the dates
histo_steps <- ggplot(data=data_no_NA) +
        geom_bar(aes(x=factor(date), y=steps), stat= "identity", fill = "blue", width = 0.5) +
        theme(axis.text.x = element_text(angle = 90)) +
        labs(x = "Date") + labs(y = "Total Steps") + labs(title = "Total Steps Taken Each Day")
histo_steps
```



Calculate and report the mean and median of the total number of steps taken per day

```
#sets the data_no_NA$steps variable as numeric
data_no_NA$steps <- as.numeric(data_no_NA$steps)
#creates a new table  from the data_no_NA data.table that finds the median and average steps grouped by date usin
g dplyr
mean_steps_table <- data_no_NA %>% group_by(date) %>% summarise(mean(steps), median(steps))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
colnames(mean_steps_table)[2] <- "MeanSteps"
colnames(mean_steps_table)[3] <- "MedianSteps"
mean_steps_table
```

```
## # A tibble: 53 x 3
##    date       MeanSteps MedianSteps
##    <date>         <dbl>       <dbl>
##  1 2012-10-02     0.438           0
##  2 2012-10-03    39.4             0
##  3 2012-10-04    42.1             0
##  4 2012-10-05    46.2             0
##  5 2012-10-06    53.5             0
##  6 2012-10-07    38.2             0
##  7 2012-10-09    44.5             0
##  8 2012-10-10    34.4             0
##  9 2012-10-11    35.8             0
## 10 2012-10-12    60.4             0
## # ... with 43 more rows
```

```
#strangely, the median number of steps is in fact 0 -
#double check median steps:
median(data_no_NA$steps)
```

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

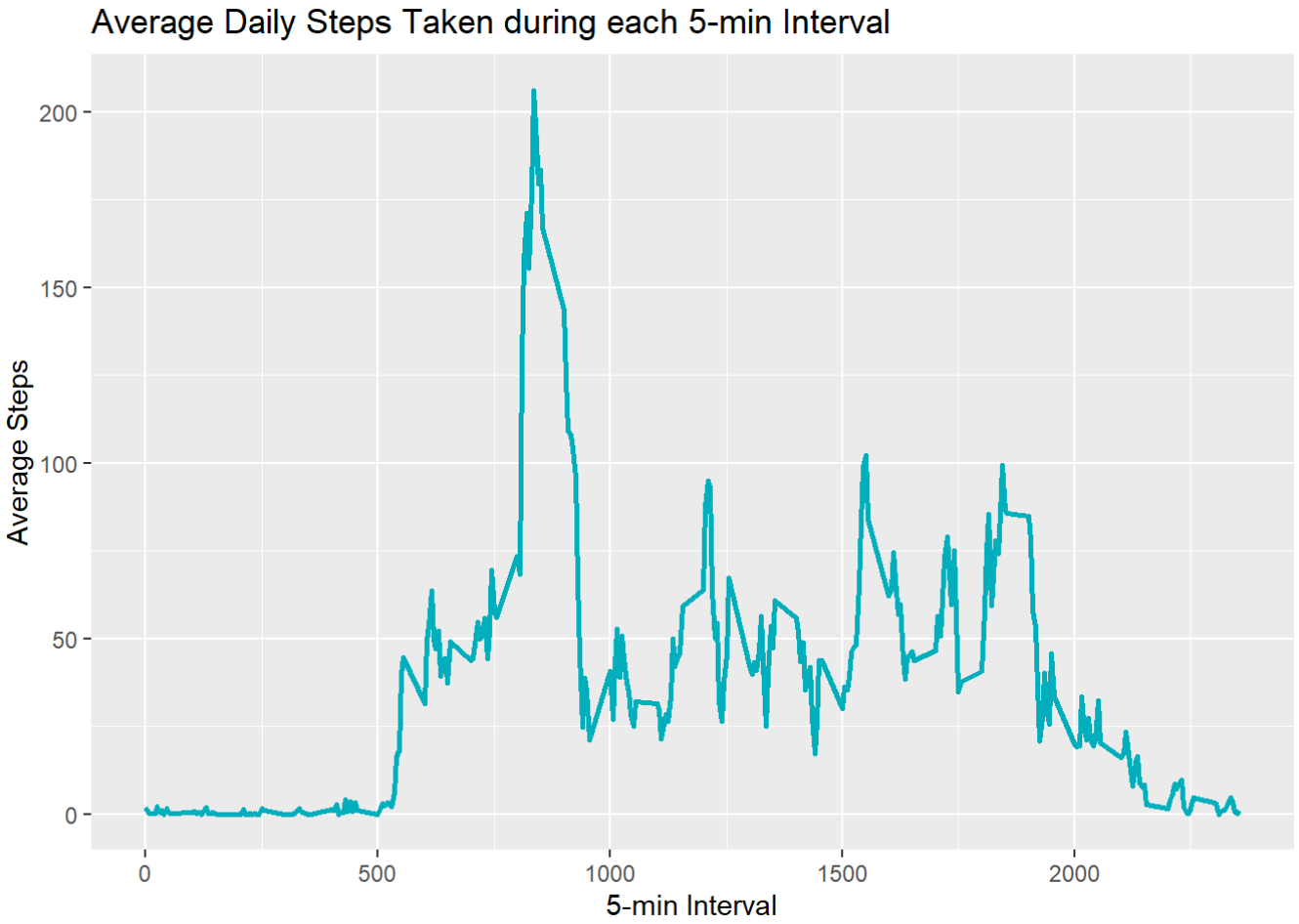
##What is the average daily activity pattern?

Make a time series plot of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all days (y-axis)

```
#creates a new table  from the data_no_NA data.table that summerises the average steps grouped by interval using
dplyr - for next graph
avg_daily_activity_pattern <- data_no_NA %>% group_by(interval) %>% summarise(mean(steps))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
colnames(avg_daily_activity_pattern)[2] <- "MeanSteps"

#uses avg_daily_activity_pattern data.table to plot a time line the average number of steps (y-axis) for each int
erval (x-axis)
time_plot <- ggplot(data=avg_daily_activity_pattern) +
        geom_line(aes(x=interval, y=MeanSteps), color = "#00AFBB", size = 1) +
        labs(x = "5-min Interval") + labs(y = "Average Steps") + labs(title = "Average Daily Steps Taken during e
ach 5-min Interval ")
time_plot
```



Which 5-minute interval, on average across all the days in the dataset, contains the maximum number of steps?

```
#creates a new table that shows (sorted_avg_daily_activity_pattern) sorted by MeanStep in decreasing order
sorted_avg_daily_activity_pattern <- avg_daily_activity_pattern[order(-avg_daily_activity_pattern$MeanSteps),]
sorted_avg_daily_activity_pattern
```

```
## # A tibble: 288 x 2
##    interval MeanSteps
##       <int>     <dbl>
```

```
##  1      835      206.
##  2      840      196.
##  3      850      183.
##  4      845      180.
##  5      830      177.
##  6      820      171.
##  7      855      167.
##  8      815      158.
##  9      825      155.
## 10      900      143.
## # ... with 278 more rows
```

```
#the interval with the highest average number of steps is:
sorted_avg_daily_activity_pattern[1,1]
```

```
## # A tibble: 1 x 1
##   interval
##      <int>
## 1      835
```

##Imputing missing values

Note that there are a number of days/intervals where there are missing values (coded as NA). The presence of missing days may introduce bias into some calculations or summaries of the data.

Calculate and report the total number of missing values in the dataset (i.e. the total number of rows with NAs)

```
#lines below provide the number of NAs in each of the interval, date, and  steps columns
sum(is.na(data$interval))
```

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

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

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

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

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

Devise a strategy for filling in all of the missing values in the dataset. The strategy does not need to be sophisticated. For example, you could use the mean/median for that day, or the mean for that 5-minute interval, etc. Create a new dataset that is equal to the original dataset but with the missing data filled in.

```
data_noZ <- data

#before:
head(data_noZ)
```

```
##    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
```

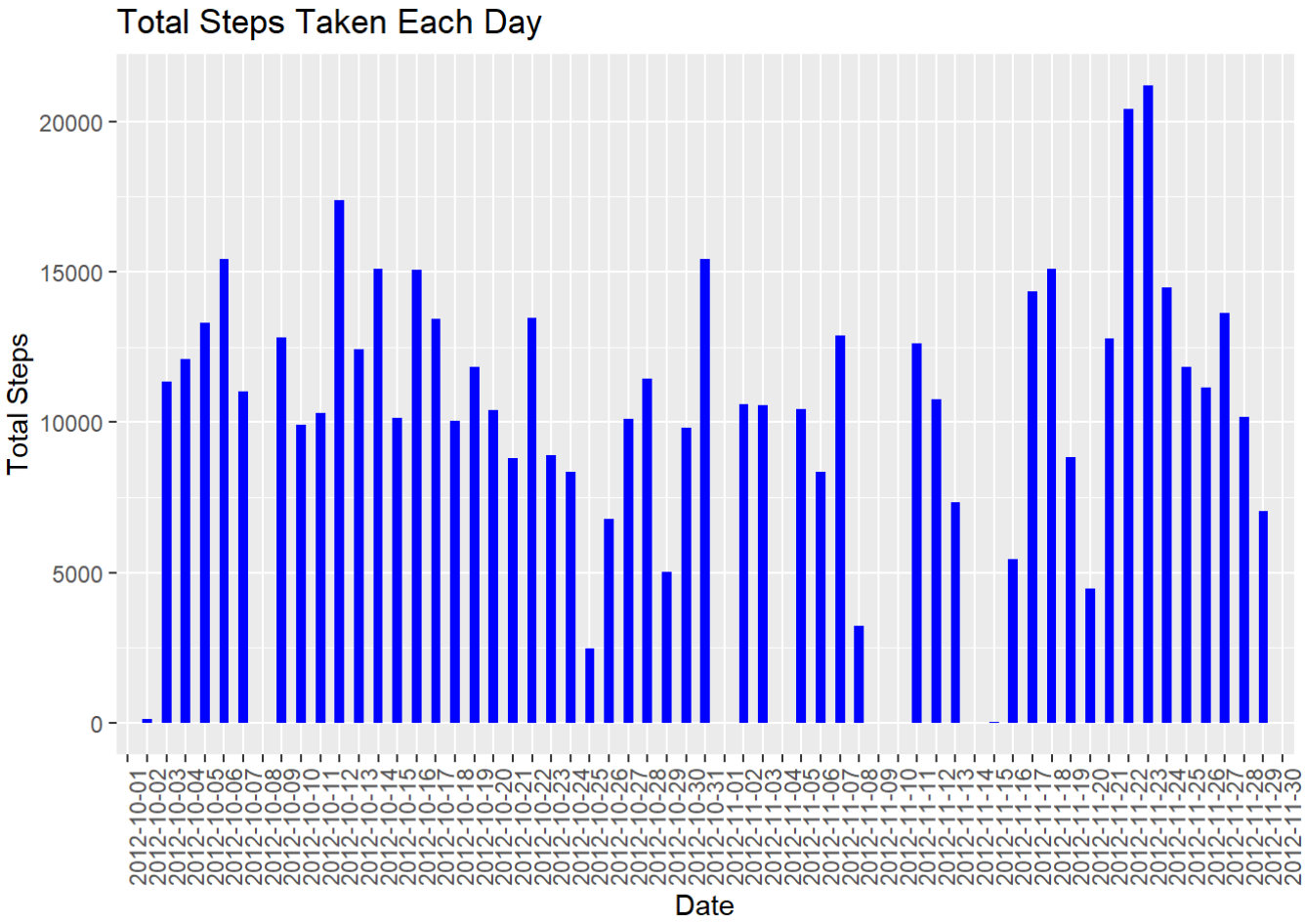
```
#replace all NAs with 0
data_noZ[is.na(data_noZ)] <- 0

#after:
head(data_noZ)
```

```
##    steps       date interval
## 1:     0 2012-10-01        0
## 2:     0 2012-10-01        5
```

```
## 3:     0 2012-10-01       10
## 4:     0 2012-10-01       15
## 5:     0 2012-10-01       20
## 6:     0 2012-10-01       25
```

Make a histogram of the total number of steps taken each day and Calculate and report the mean and median total number of steps taken per day.

```
histo_steps2 <- ggplot(data=data_noZ) +
        geom_bar(aes(x=factor(date), y=steps), stat= "identity", fill = "blue", width = 0.5) +
        theme(axis.text.x = element_text(angle = 90)) +
        labs(x = "Date") + labs(y = "Total Steps") + labs(title = "Total Steps Taken Each Day")
histo_steps2
```



Do these values differ from the estimates from the first part of the assignment? What is the impact of imputing missing data on the estimates of the total daily number of steps?

```
#sets the data_no_NA$steps variable as numeric
data_noZ$steps <- as.numeric(data_noZ$steps)
#creates a new table  from the data_no_NA data.table that finds the median and average steps grouped by date usin
g dplyr
mean_steps_table_noZ <- data_noZ %>% group_by(date) %>% summarise(mean(steps), median(steps))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
colnames(mean_steps_table_noZ)[2] <- "MeanSteps"
colnames(mean_steps_table_noZ)[3] <- "MedianSteps"
mean_steps_table_noZ
```

```
## # A tibble: 61 x 3
##    date       MeanSteps MedianSteps
##    <date>         <dbl>       <dbl>
##  1 2012-10-01     0               0
##  2 2012-10-02     0.438           0
##  3 2012-10-03    39.4             0
##  4 2012-10-04    42.1             0
##  5 2012-10-05    46.2             0
##  6 2012-10-06    53.5             0
##  7 2012-10-07    38.2             0
##  8 2012-10-08     0               0
##  9 2012-10-09    44.5             0
## 10 2012-10-10    34.4             0
## # ... with 51 more rows
```

```
#compare before and after:
head(mean_steps_table)
```

```
## # A tibble: 6 x 3
##   date       MeanSteps MedianSteps
##   <date>         <dbl>       <dbl>
## 1 2012-10-02     0.438           0
## 2 2012-10-03    39.4             0
## 3 2012-10-04    42.1             0
## 4 2012-10-05    46.2             0
## 5 2012-10-06    53.5             0
## 6 2012-10-07    38.2             0
```

```
head(mean_steps_table_noZ)
```

```
## # A tibble: 6 x 3
##   date       MeanSteps MedianSteps
##   <date>         <dbl>       <dbl>
## 1 2012-10-01     0               0
## 2 2012-10-02     0.438           0
## 3 2012-10-03    39.4             0
## 4 2012-10-04    42.1             0
## 5 2012-10-05    46.2             0
## 6 2012-10-06    53.5             0
```

```
#just removed the 0s
```

##Are there differences in activity patterns between weekdays and weekends?

For this part the weekdays() weekdays() function may be of some help here. Use the dataset with the filled-in missing values for this part.

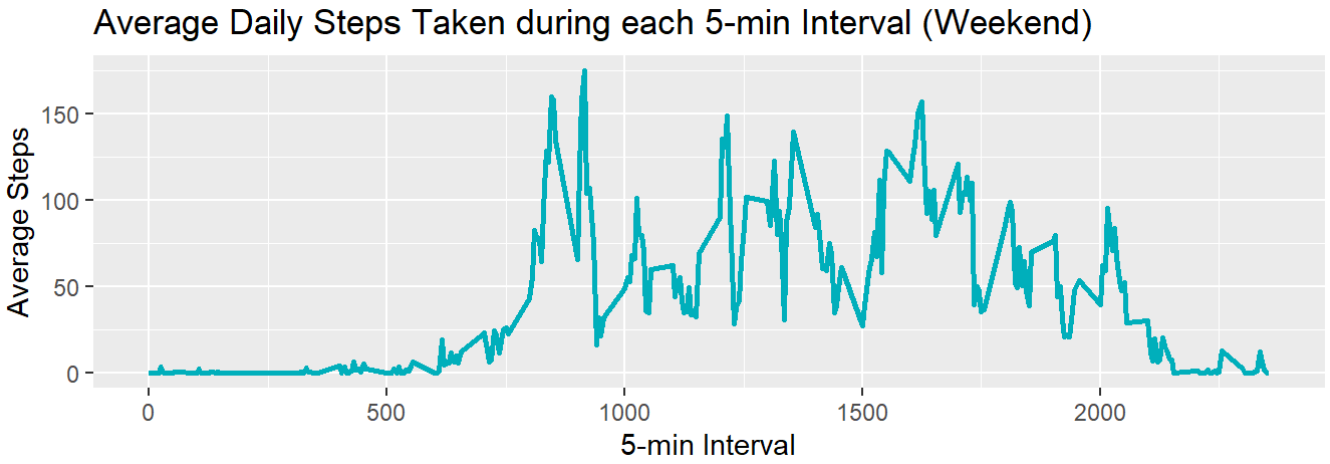
Create a new factor variable in the dataset with two levels – “weekday” and “weekend” indicating whether a given date is a weekday or weekend day.

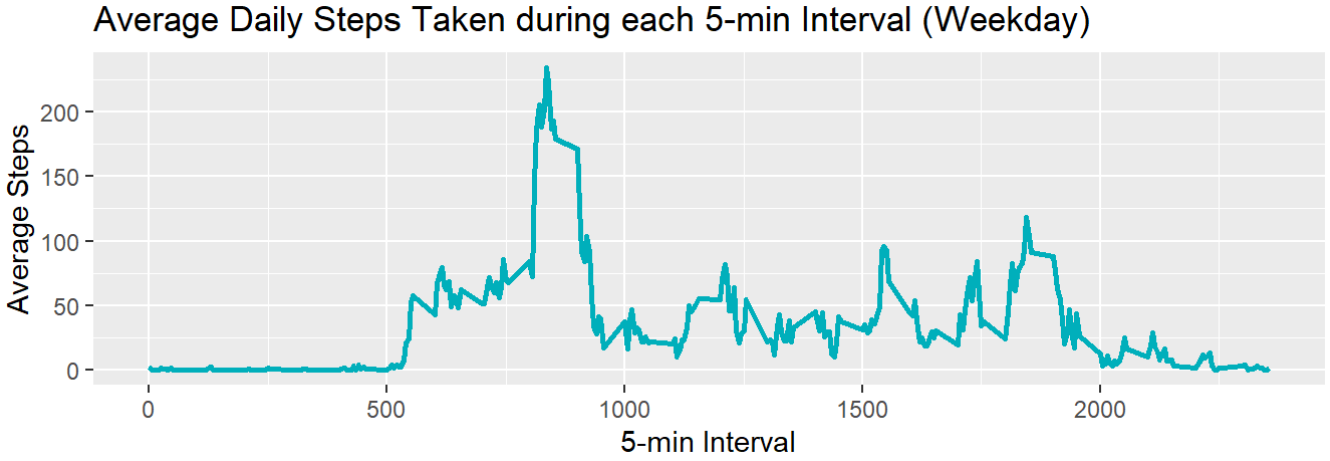
```
#create a new column in data_no_NA called "weekday" and use it to show which date was on what day
data_no_NA$weekday <- weekdays(data_no_NA$date)
#create another column in data_no_NA called "weekend". Test each value in weekday - it weekday is a saturday or s
unday, then weekend is 1, else 0
data_no_NA$weekend <- ifelse(data_no_NA$weekday == "Saturday" | data_no_NA$weekday == "Sunday", "weekend", "weekd
ay")
```

Make a panel plot containing a time series plot of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all weekday days or weekend days (y-axis).

```
#creates two new data tables - one for weekdays and the other weekends
data_no_NA_weekend <- subset(data_no_NA, data_no_NA$weekday == "Saturday" | data_no_NA$weekday == "Sunday")
data_no_NA_weekday <- subset(data_no_NA, !data_no_NA$weekday == "Saturday" & !data_no_NA$weekday == "Sunday")
```

```
####Weekend
avg_daily_activity_pattern_weekend <- data_no_NA_weekend %>% group_by(interval) %>% summarise(mean(steps))

colnames(avg_daily_activity_pattern_weekend)[2] <- "MeanSteps"

time_plot_weekend <- ggplot(data=avg_daily_activity_pattern_weekend) +
        geom_line(aes(x=interval, y=MeanSteps), color = "#00AFBB", size = 1) +
        labs(x = "5-min Interval") + labs(y = "Average Steps") + labs(title = "Average Daily Steps Taken during e
ach 5-min Interval (Weekend)")


####Weekday
avg_daily_activity_pattern_weekday <- data_no_NA_weekday %>% group_by(interval) %>% summarise(mean(steps))
colnames(avg_daily_activity_pattern_weekday)[2] <- "MeanSteps"

time_plot_weekday <- ggplot(data=avg_daily_activity_pattern_weekday) +
        geom_line(aes(x=interval, y=MeanSteps), color = "#00AFBB", size = 1) +
        labs(x = "5-min Interval") + labs(y = "Average Steps") + labs(title = "Average Daily Steps Taken during e
ach 5-min Interval (Weekday)")


source("http://peterhaschke.com/Code/multiplot.R")
multiplot(time_plot_weekend, time_plot_weekday, cols=1)
```





##Finito