Reproducible Research Course Project 1

Kalli Buchanan

24/11/2020

Loading and Preprocessing the Data

Download data from provided url and load as dataframe.

```
library(readr)

if(!file.exists(".data")){dir.create("./data")}
```

```
## Warning in dir.create("./data"): './data' already exists
```

```
fileUrl <- "https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip"
download.file(fileUrl, destfile = "./data/activity_monitoring.csv", method = "curl")
activityData <- read_csv("data/activity_monitoring.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   steps = col_double(),
##   date = col_date(format = ""),
##   interval = col_double()
## )
```

Data is already processed, so no need to process further.

```
head(activityData)
```

```
## # A tibble: 6 x 3
##   steps date       interval
##   <dbl> <date>        <dbl>
## 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
```

Let's take a quick look at what the data is comprised of.

```
summary(activityData)
```

```
##      steps             date               interval
##  Min.   :  0.00   Min.   :2012-10-01   Min.   :   0.0
##  1st Qu.:  0.00   1st Qu.:2012-10-16   1st Qu.: 588.8
##  Median :  0.00   Median :2012-10-31   Median :1177.5
##  Mean   : 37.38   Mean   :2012-10-31   Mean   :1177.5
##  3rd Qu.: 12.00   3rd Qu.:2012-11-15   3rd Qu.:1766.2
##  Max.   :806.00   Max.   :2012-11-30   Max.   :2355.0
##  NA's   :2304
```

Mean total number of steps taken

Calculate the total number of steps taken per day and create new data.frame to hold this information

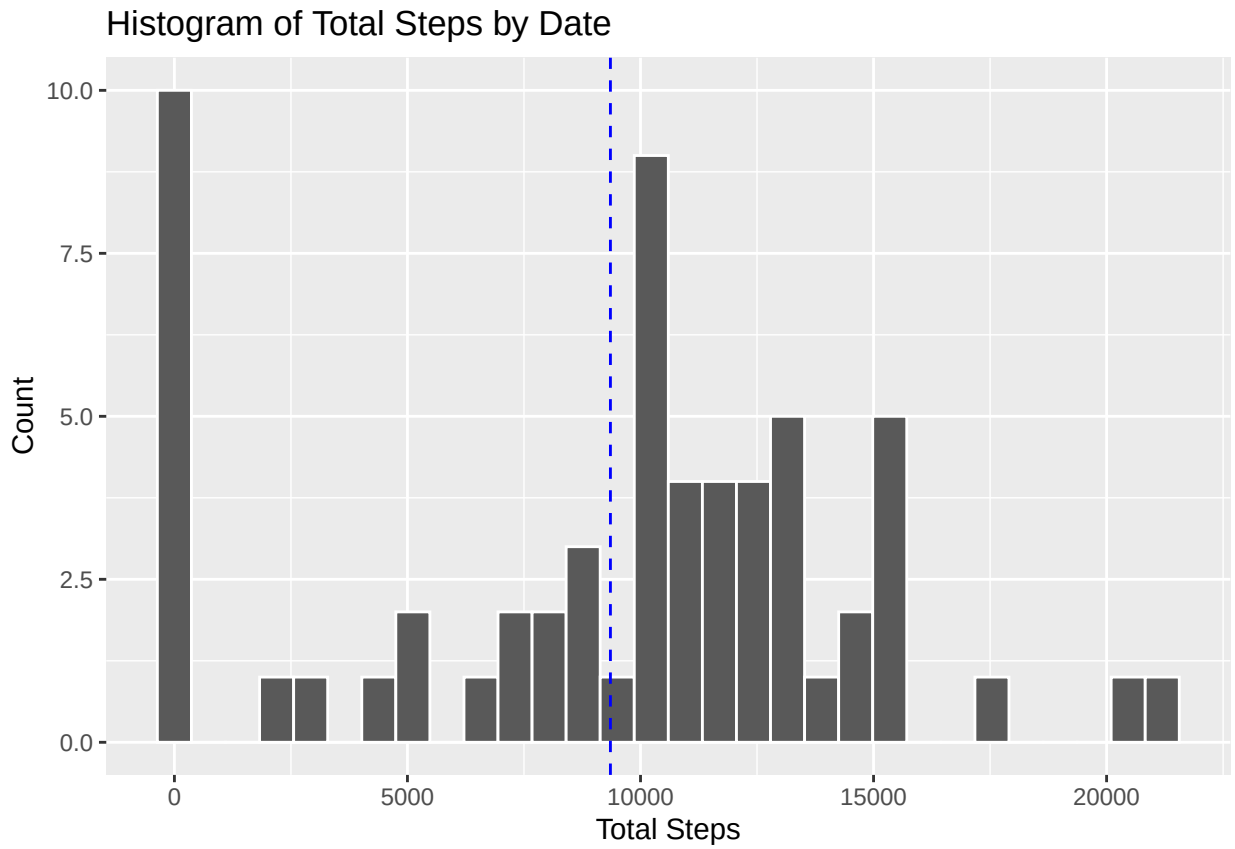
```
stepsData <- with(activityData, data.frame(date = unique(date),
                                           total.steps = tapply(steps, date,
                                                                sum, na.rm = TRUE)))
head(stepsData)
```

```
##                  date total.steps
## 2012-10-01 2012-10-01           0
## 2012-10-02 2012-10-02         126
## 2012-10-03 2012-10-03       11352
## 2012-10-04 2012-10-04       12116
## 2012-10-05 2012-10-05       13294
## 2012-10-06 2012-10-06       15420
```

Histogram of the total amount of steps by each day.

```
library(ggplot2)
g <- ggplot(stepsData, aes(total.steps))
g + geom_histogram(colour = "white") +
  geom_vline(aes(xintercept=mean(total.steps)), colour="blue", lty="dashed", size=0.5) +
  labs(title = "Histogram of Total Steps by Date",
       x = "Total Steps",
       y = "Count")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Find the mean and median of total number of steps taken per day.

```
stepsMean <- round(mean(stepsData$total.steps), digits = 2)
stepsMedian <- round(median(stepsData$total.steps), digits = 2)
```

The mean is **9354.23** and the median is **10395**.

Average Daily Activity Pattern

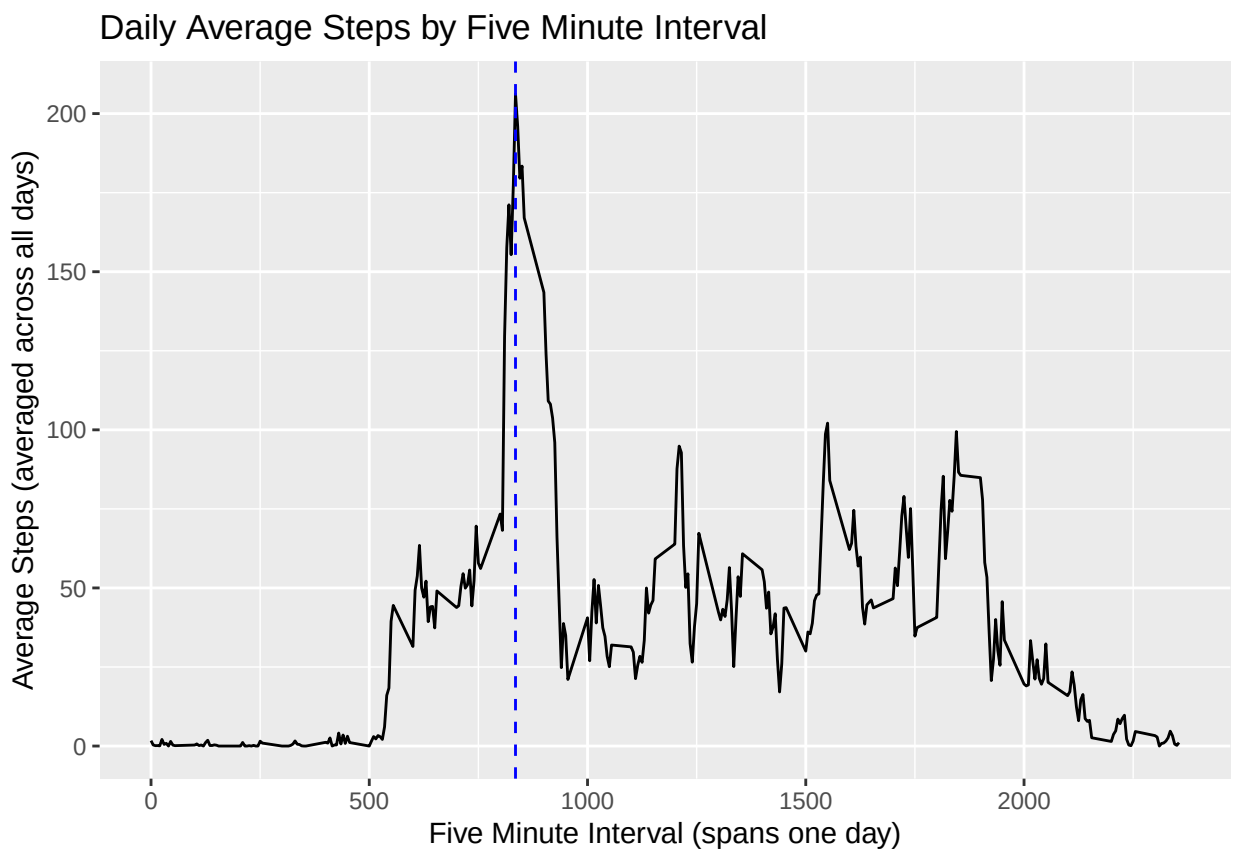
Make a new data.frame which holds unique intervals with the number of steps averaged across all days.

```
intervalData <- with(activityData, data.frame(interval = unique(interval),
                                              average.steps = tapply(steps, interval,
                                                                     mean, na.rm = TRUE)))
head(intervalData)
```

```
##    interval average.steps
## 0         0     1.7169811
## 5         5     0.3396226
## 10       10     0.1320755
## 15       15     0.1509434
## 20       20     0.0754717
## 25       25     2.0943396
```

View these results using a time series plot. Dashed vertical line shows the interval with the highest average steps.

```
ts <- ggplot(intervalData, aes(interval, average.steps))
ts + geom_line() +
  geom_vline(aes(xintercept = interval[average.steps == max(average.steps)]), col = "blue", lty = "dash
  labs(title = "Daily Average Steps by Five Minute Interval",
       x = "Five Minute Interval (spans one day)",
       y = "Average Steps (averaged across all days)")
```



Find the interval with the highest average steps across all days.

```
maxInterval <- with(intervalData, interval[average.steps == max(average.steps)])
maxAvgSteps <- with(intervalData, round(average.steps[interval == maxInterval], digits = 2))
```

The interval **835**, has the highest average steps (**206.17**).

Impute Missing Values

Find number of missing values in the data set.

```
sum(is.na(activityData))
```

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

Fill in missing values in data set using an average of the previous and next day steps for that specific interval.

```
imputedData = activityData

for (i in 1:dim(imputedData)[1]) {
  if (is.na(imputedData$steps[i])){
    before <- ifelse(imputedData$date[i] == "2012-10-01", NA,
                     imputedData$steps[imputedData$date == imputedData$date[i]-1 &
                                         imputedData$interval == imputedData$interval[i]])
    after <- ifelse(imputedData$date[i] == "2012-11-30", NA,
                    imputedData$steps[imputedData$date == imputedData$date[i]+1
                                      & imputedData$interval == imputedData$interval[i]])
    newSteps <- mean(c(before,after), na.rm = TRUE)
    imputedData$steps[i] = newSteps
  }
}
sum(is.na(imputedData))
```

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

This shows that all missing values have been imputed.

Re-vist finding from original calculations when missing values were ignored.

First create a new data frame with total steps per day.

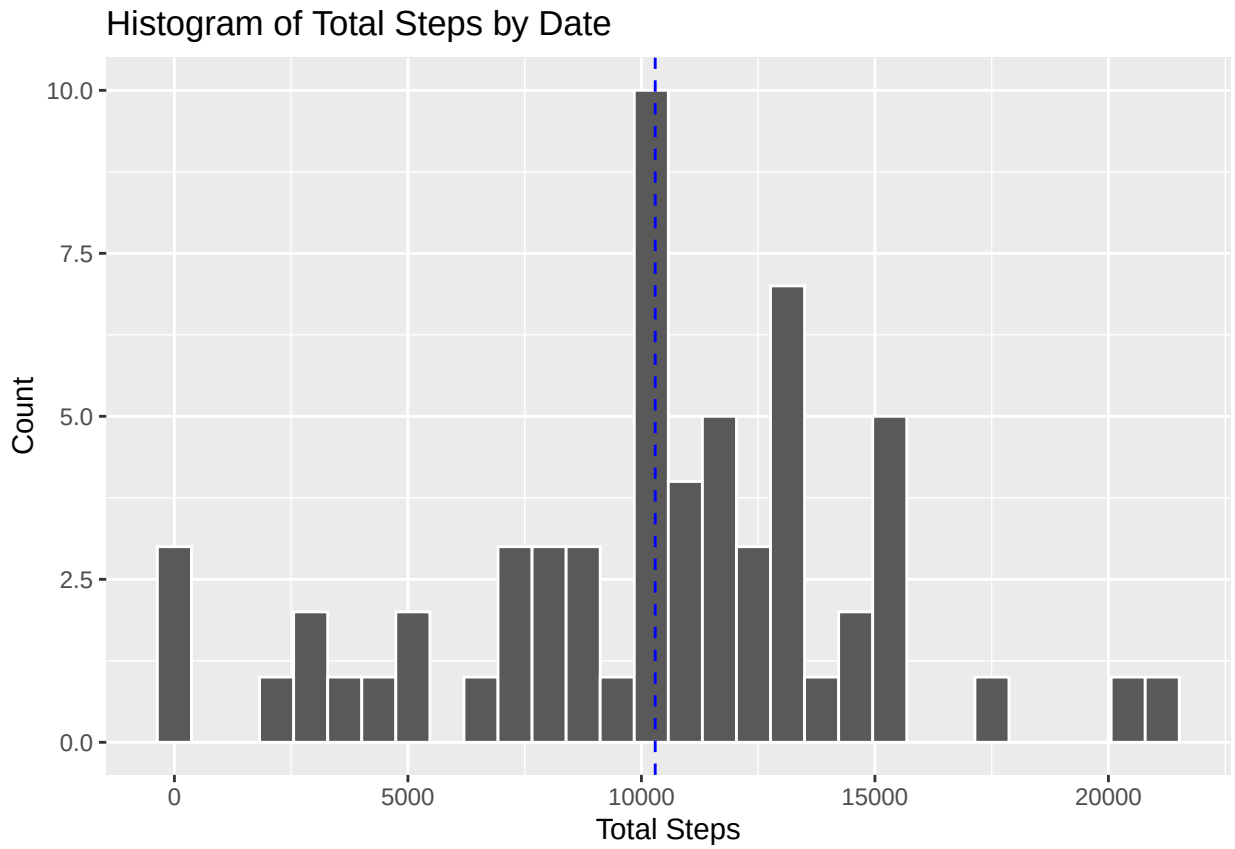
```
imputedSteps <- with(imputedData, data.frame(date = unique(date),
                                             total.steps = tapply(steps, date,
                                                                  sum, na.rm = FALSE)))
head(imputedSteps)
```

```
##                  date total.steps
## 2012-10-01 2012-10-01         126
## 2012-10-02 2012-10-02         126
## 2012-10-03 2012-10-03       11352
## 2012-10-04 2012-10-04       12116
## 2012-10-05 2012-10-05       13294
## 2012-10-06 2012-10-06       15420
```

Histogram of the total steps per day.

```
ig <- ggplot(imputedSteps, aes(total.steps))
ig + geom_histogram(colour = "white") +
  geom_vline(aes(xintercept=mean(total.steps)), colour="blue", lty="dashed", size=0.5) +
  labs(title = "Histogram of Total Steps by Date",
       x = "Total Steps",
       y = "Count")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Find the mean and median of total number of steps taken per day.

```
imputedMean <- round(mean(imputedSteps$total.steps), digits = 2)
imputedMedian <- round(median(imputedSteps$total.steps), digits = 2)
```

The mean is **10295.52** and the median is **10571**. These are both larger than the original values by **941.29** and **176**, respectively.

Weekdays and Weekends Activity Patterns

Add new factor variable to data set to indicate whether the date is during the week or weekend.

```
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
```

```
weekendday <- c("Saturday", "Sunday")
weekData <- imputedData %>%
  mutate(date.type = ifelse(weekdays(date) %in% weekendday, "weekend", "weekday"), date.type = as.factor
summary(weekData)
```

```
##      steps             date               interval        date.type
##  Min.   :  0.00   Min.   :2012-10-01   Min.   :   0.0   weekday:12960
##  1st Qu.:  0.00   1st Qu.:2012-10-16   1st Qu.: 588.8   weekend: 4608
##  Median :  0.00   Median :2012-10-31   Median :1177.5
##  Mean   : 35.75   Mean   :2012-10-31   Mean   :1177.5
##  3rd Qu.: 12.12   3rd Qu.:2012-11-15   3rd Qu.:1766.2
##  Max.   :806.00   Max.   :2012-11-30   Max.   :2355.0
```

Make two new data.frame which holds unique intervals with the number of steps averaged across all days; one for weekdays, and one for weekends. Join these tables based on interval.

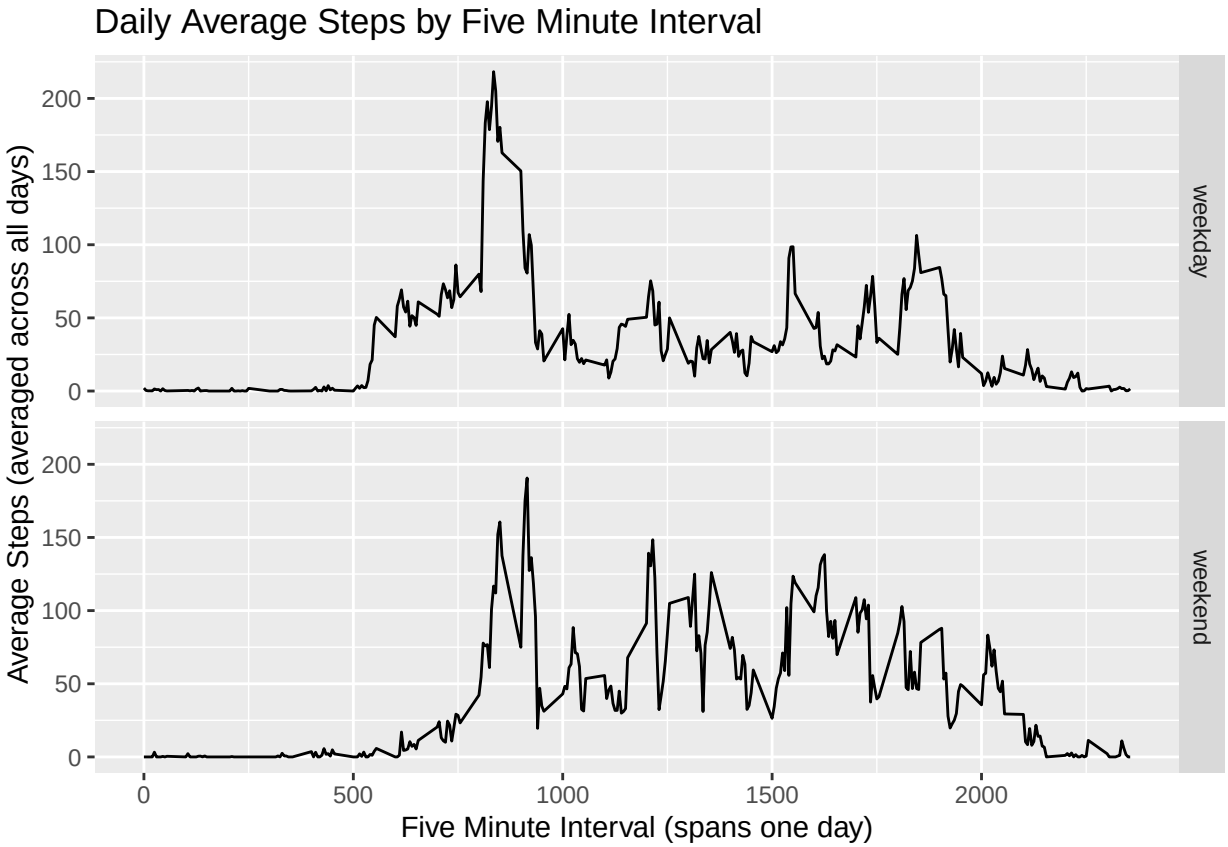
```
library(tidyr)
weekdayData <- with(weekData[weekData$date.type == "weekday",], data.frame(interval = unique(interval),
                                                 average.steps.weekday = tapply(steps, interval,
                                                                                mean)))
weekendData <- with(weekData[weekData$date.type == "weekend",], data.frame(interval = unique(interval),
                                                 average.steps.weekend = tapply(steps, interval,
                                                                                mean)))
weekInterval <- merge(weekdayData, weekendData) %>%
  pivot_longer(!interval, names_to = "date.type", values_to = "average.steps", names_prefix = "average.s
  mutate(date.type = as.factor(date.type))

head(weekInterval)
```

```
## # A tibble: 6 x 3
##   interval date.type average.steps
##      <dbl> <fct>             <dbl>
## 1        0 weekday           2.02
## 2        0 weekend           0
## 3        5 weekday           0.4
## 4        5 weekend           0
## 5       10 weekday           0.156
## 6       10 weekend           0
```

View these results using a time series plot.

```
wd <- ggplot(weekInterval, aes(interval, average.steps))
wd + facet_grid(date.type~.) +
  geom_line() +
  labs(title = "Daily Average Steps by Five Minute Interval",
       x = "Five Minute Interval (spans one day)",
       y = "Average Steps (averaged across all days)")
```


Daily Average Steps by Five Minute Interval
Average Steps (averaged across all days)
200
150
100
50
0
weekday
weekend
0
500
1000
1500
2000
Five Minute Interval (spans one day)