Reproducible Research: Peer Assessment 1

Assignment Info

- Filename: PA1_template.Rmd
- Name: Walter Yu
- Date: July 2020

Introduction

This markdown file is an analysis and visualization of UCI human activity recognition dataset. Source data for this assignment is an aggregated dataset from the original available here (https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones).

This assignment is completed for the JHU Coursera Data Science Program, which is a 10 course certification. More info about this program is available here (https://www.coursera.org/specializations/jhu-data-science).

HTML File/Template

The HTML file for this assignment was generated with R Studio Cloud (https://rstudio.cloud) due to dplyr package installation errors on my local machine; as a result, file was run and report rendered there instead.

This markdown project template is based on a fork of the course assignment Github repo available here (https://rstudio.cloud).

Notes

1. All code included to show data import and plot
2. Full dataset was imported, then subset for plot
3. Plotting systems used per assignment instructions

Loading and preprocessing the data

Part 1: Import data per assignment instructions

Processing Steps: * Loaded data with read.csv function per instructions * File was unzipped, then read into program

Analysis: * Used Head, names and summary functions to review the data * Data has null values (2308 rows) and median steps of zero

Part 2: Remove null values

Processing Steps: * Used na.omit function to remove null values * Assigned results to a new variable

Analysis: * Used head, names and summary functions to review the data * Data has null values (2308 rows) and median steps of zero * Summary function indicated mean/median steps did not change

```
# part 1: import data per assignment instructions
# source: assignment instructions
activity <- read.csv("activity.csv")

# review dataset
# https://www.statmethods.net/stats/descriptives.html
head(activity)
```

```
##   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
```

```
names(activity)
```

```
## [1] "steps"    "date"     "interval"
```

```
summary(activity)
```

```
##      steps            date              interval
##  Min.   :  0.00   Length:17568       Min.   :   0.0
##  1st Qu.:  0.00   Class :character   1st Qu.: 588.8
##  Median :  0.00   Mode  :character   Median :1177.5
##  Mean   : 37.38                      Mean   :1177.5
##  3rd Qu.: 12.00                      3rd Qu.:1766.2
##  Max.   :806.00                      Max.   :2355.0
##  NA's   :2304
```

```
# part 2: remove null values
# source: https://www.statmethods.net/input/missingdata.html
activity_omit <- na.omit(activity)

# verify results
# https://www.statmethods.net/stats/descriptives.html
head(activity_omit)
```

```
##     steps       date interval
## 289     0 2012-10-02        0
## 290     0 2012-10-02        5
## 291     0 2012-10-02       10
## 292     0 2012-10-02       15
## 293     0 2012-10-02       20
## 294     0 2012-10-02       25
```

```
names(activity_omit)
```

```
## [1] "steps"    "date"     "interval"
```

```
summary(activity_omit)
```

```
##      steps            date              interval     
##  Min.   :  0.00   Length:15264       Min.   :   0.0  
##  1st Qu.:  0.00   Class :character   1st Qu.: 588.8  
##  Median :  0.00   Mode  :character   Median :1177.5  
##  Mean   : 37.38                      Mean   :1177.5  
##  3rd Qu.: 12.00                      3rd Qu.:1766.2  
##  Max.   :806.00                      Max.   :2355.0
```

What is mean total number of steps taken per day?

Part 1: Calculate sum, then create histogram

Processing Steps: * Used aggreage function to calculate total steps/day * Assigned results to a new variable

Analysis: * Used head, names and summary functions to review the data * Verified that aggregate returned total steps per day

Plot: * Created histogram for steps/day; axis adjusted to fit data * Applied bins to visualize the data effectively

Part 2: Calculate mean/median by date

Processing Steps: * Used aggreage function to calculate mean/median steps/day * Assigned results to a new variable

Analysis: * Used head, names and summary functions to review the data * Verified that aggregate returned mean/median steps per day * Median steps/day = 0 due to large number of zero step values

```
# part 1: calculate sum, then create histogram
# aggregate by date, then apply calculations
# source: https://www.statmethods.net/management/aggregate.html
steps_sum <- aggregate(steps ~ date, activity_omit, sum)

# verify results
# https://www.statmethods.net/stats/descriptives.html
head(steps_sum)
```

```
##         date steps
## 1 2012-10-02   126
## 2 2012-10-03 11352
## 3 2012-10-04 12116
## 4 2012-10-05 13294
## 5 2012-10-06 15420
## 6 2012-10-07 11015
```

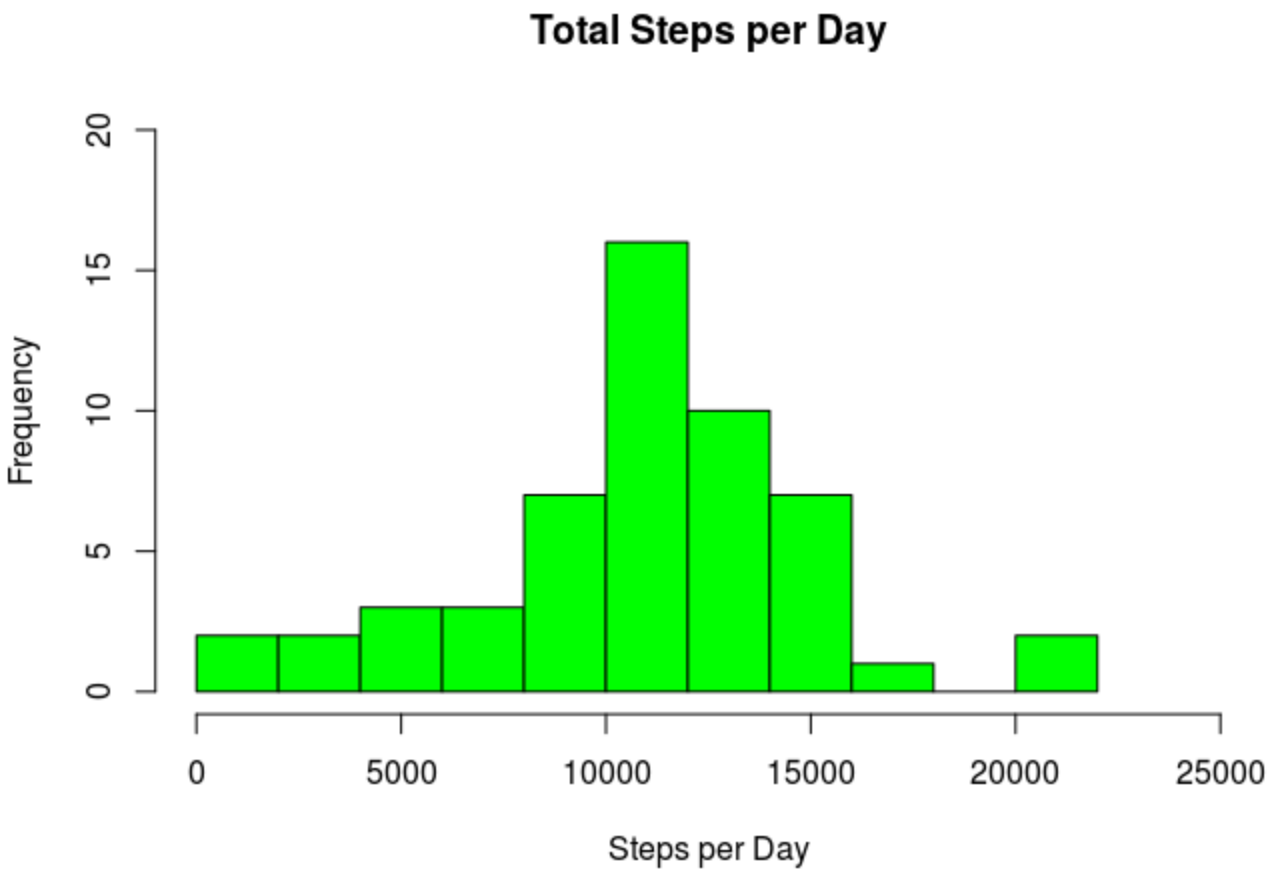
```
names(steps_sum)
```

```
## [1] "date"  "steps"
```

```
summary(steps_sum)
```

```
##      date               steps
##  Length:53          Min.   :   41
##  Class :character   1st Qu.: 8841
##  Mode  :character   Median :10765
##                     Mean   :10766
##                     3rd Qu.:13294
##                     Max.   :21194
```

```
# create histogram plot; adjust axis and bins
# source: https://www.statmethods.net/graphs/density.html
hist(steps_sum$steps, col="green",
     breaks=10, main="Total Steps per Day",
     xlab="Steps per Day", ylim=c(0,20), xlim=c(0,25000))
```




```
# part 2: calculate mean/median by date
# aggregate by date, then apply calculations
# source: https://www.statmethods.net/management/aggregate.html
steps_mean <- aggregate(steps ~ date, activity_omit, mean)

# verify results
# https://www.statmethods.net/stats/descriptives.html
head(steps_mean)
```

```
##         date    steps
## 1 2012-10-02  0.43750
## 2 2012-10-03 39.41667
## 3 2012-10-04 42.06944
## 4 2012-10-05 46.15972
## 5 2012-10-06 53.54167
## 6 2012-10-07 38.24653
```

```
names(steps_mean)
```

```
## [1] "date"  "steps"
```

```
summary(steps_mean)
```

```
##      date               steps
##  Length:53          Min.   : 0.1424
##  Class :character   1st Qu.:30.6979
##  Mode  :character   Median :37.3785
##                     Mean   :37.3826
##                     3rd Qu.:46.1597
##                     Max.   :73.5903
```

```
# aggregate by date, then apply calculations
# source: https://www.statmethods.net/management/aggregate.html
steps_median <- aggregate(steps ~ date, activity_omit, median)

# verify results
# https://www.statmethods.net/stats/descriptives.html
head(steps_median)
```

```
##         date steps
## 1 2012-10-02     0
## 2 2012-10-03     0
## 3 2012-10-04     0
## 4 2012-10-05     0
## 5 2012-10-06     0
## 6 2012-10-07     0
```

```
names(steps_median)
```

```
## [1] "date"  "steps"
```

```
summary(steps_median)
```

```
##      date               steps
##  Length:53          Min.   :0
##  Class :character   1st Qu.:0
##  Mode  :character   Median :0
##                     Mean   :0
##                     3rd Qu.:0
##                     Max.   :0
```

What is the average daily activity pattern?

Part 1: Group by interval, calculate mean and plot time series

Processing Steps: * Used dplyr/group_by to calculate mean steps/interval * Assigned results to a new variable

Analysis: * Used head, names and summary functions to review the data * Verified that mean of steps/interval differ from steps/day

Plot: * Created time series plot; add axis labels and line color

Notes: * Assumed instructions were for steps/interval across ALL DAYS * So, mean was calculated per interval for entire data set * NOT group by interval, then calculate mean by date

Part 2: Find interval with maximum steps across all days

Processing Steps: * Used which function to subset for maximum steps/interval * Assigned results to a new variable

Analysis: * Interval 835 had the maximum steps/interval.

```
# part 1: group by interval, calculate mean and plot time series
# group by interval, then calculate mean steps for each group
# source: https://datacarpentry.org/R-genomics/04-dplyr.html
install.packages("dplyr")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/4.0'
## (as 'lib' is unspecified)
```

```
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
```

```
steps_avg_interval <- activity_omit %>%
    group_by(interval) %>%
    summarize(steps_avg = mean(steps))
```

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

```
# verify results
# https://www.statmethods.net/stats/descriptives.html
head(steps_avg_interval)
```

```
## # A tibble: 6 x 2
##   interval steps_avg
##      <int>     <dbl>
## 1        0    1.72
## 2        5    0.340
## 3       10    0.132
## 4       15    0.151
## 5       20    0.0755
## 6       25    2.09
```

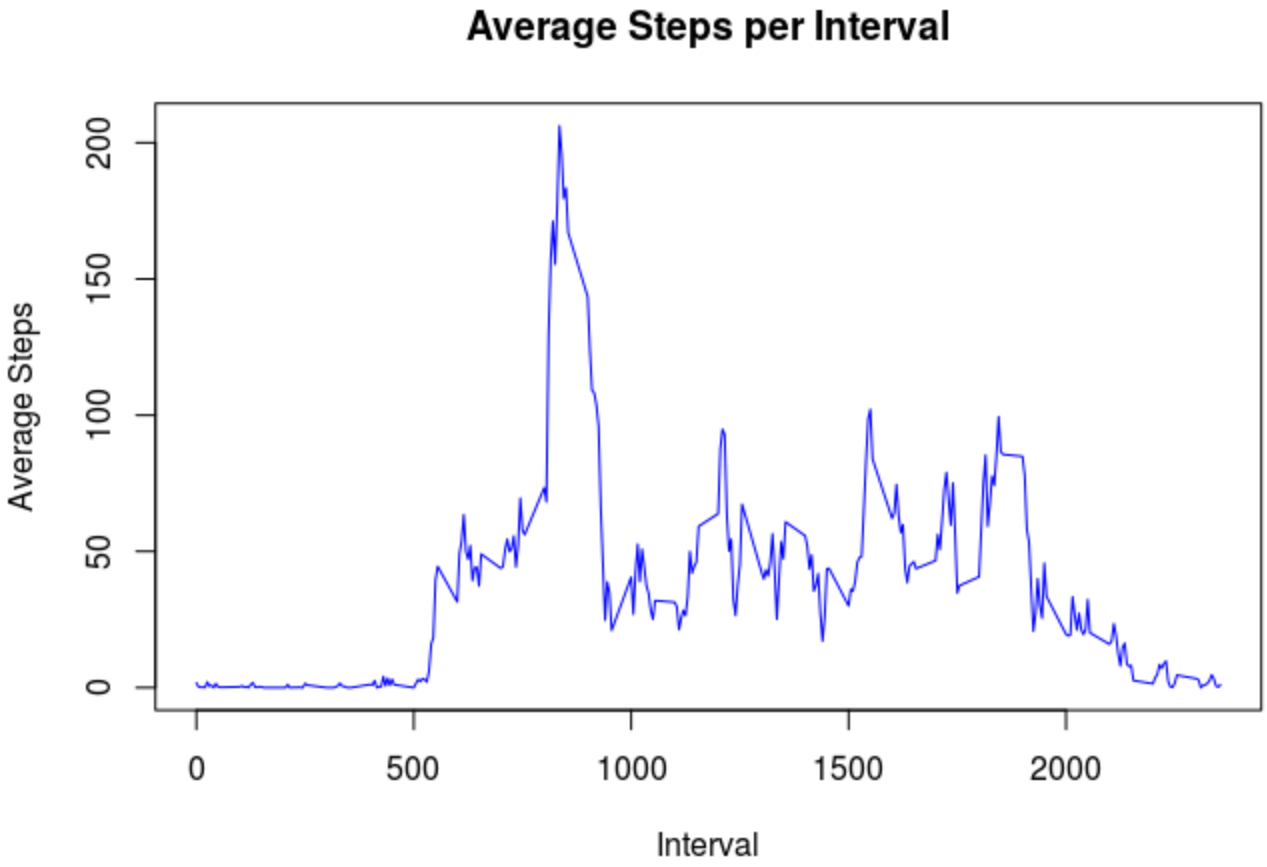
```
names(steps_avg_interval)
```

```
## [1] "interval"  "steps_avg"
```

```
summary(steps_avg_interval)
```

```
##     interval        steps_avg
##  Min.   :   0.0   Min.   :  0.000
##  1st Qu.: 588.8   1st Qu.:  2.486
##  Median :1177.5   Median : 34.113
##  Mean   :1177.5   Mean   : 37.383
##  3rd Qu.:1766.2   3rd Qu.: 52.835
##  Max.   :2355.0   Max.   :206.170
```

```
# create time series plot; use date list and sum of steps
# source: https://www.datamentor.io/r-programming/plot-function/
plot(steps_avg_interval,
     main="Average Steps per Interval",
     xlab="Interval",
     ylab="Average Steps",
     col="blue",
     type="l")
```




```
# part 2: find interval with maximum steps across all days
# source: https://stackoverflow.com/questions/19449615/how-to-extract-the-row-with-min-or-max-values
max_interval <- steps_avg_interval[which.max(steps_avg_interval$steps_avg),]

# return row with max interval
print("Subset for maximum interval and return result.")
```

```
## [1] "Subset for maximum interval and return result."
```

```
max_interval
```

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

Imputing missing values

Part 1: Return count of rows with null values

Processing Steps: * Used is.na function to subset for records with null * Assigned results to a new variable

Analysis: * Verified that row count (2308) was same as raw data/summary

Part 2-3: Create new dataset with imputed null values

Processing Steps: * Used simputation/impute_lm to impute null values * Replaced null values with linear model per interval * Assigned dataset to new variable

Analysis: * Imputation did not change mean/median * Change did not occur since linear model filled matching values * However, total steps increased since filled values add to total

Notes: * Numerous attempts were made with dplyr/group_by which failed * Code left to document previous attempts * As a result, simputation package was used to yield result

Part 4: calculate sum, then create histogram

Processing Steps: * Used aggreage function to calculate total steps/day * Assigned results to a new variable

Plot: * Created time series plot; add axis labels and line color * Total steps increased since filled values add to total

Analysis: * Imputation did not change mean/median * Change did not occur since linear model filled matching values * However, total steps increased since filled values add to total

```
# part 1: return count of rows with null values
# https://stackoverflow.com/questions/7980622/subset-of-rows-containing-na-missing-values-in-a-chosen-colum
n-of-a-data-frame
activity_na <- activity[is.na(activity),]

# return row with max interval
print("Subset for records with null values and return count.")
```

```
## [1] "Subset for records with null values and return count."
```

```
nrow(activity_na)
```

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

```
# part 2-3: create new dataset with imputed null values
# replace null with mean value by date
# source: https://datascience.stackexchange.com/questions/14065/imputing-missing-values-by-mean-by-id-column-in-r
# https://stackoverflow.com/questions/27207162/fill-in-na-based-on-the-last-non-na-value-for-each-group-in-r
# library(zoo)
# activity_imp %>%
#     group_by(date) %>%
#     mutate(step=zoo::na.locf(steps))
#     transmute(steps=na.locf(steps, na.rm=FALSE))
#     mutate(steps=ifelse(is.na(steps),mean(steps, na.rm=TRUE),steps))

# source: https://stackoverflow.com/questions/21714867/replace-na-in-a-dplyr-chain
# library(data.table)
# names(activity_imp)
# activity_imp[, steps := ifelse(
#   is.na(steps),mean(steps,na.rm=TRUE), steps), by=date]

# source: https://github.com/decisionpatterns/tidyimpute/issues/5
# activity_imp %>%
#   group_by(date) %>%
#   group_modify(~ impute_median(.x, steps)) %>%
#   ungroup()

# source: https://tidyr.tidyverse.org/reference/fill.html
# install.packages("tidyverse")
# library(tidyverse)
# activity_imp %>%
#   group_by(date) %>%
#   fill(steps, .direction="downup") %>%
#   ungroup()

# source: https://cran.r-project.org/web/packages/simputation/vignettes/intro.html
install.packages('simputation')
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/4.0'
## (as 'lib' is unspecified)
```

```
library(simputation)
activity_imp <- impute_lm(activity, steps ~ interval)

# verify results
# https://www.statmethods.net/stats/descriptives.html
nrow(activity_imp)
```

```
## [1] 17568
```

```
head(activity_imp)
```

```
##      steps       date interval
## 1 29.55387 2012-10-01        0
## 2 29.58711 2012-10-01        5
## 3 29.62036 2012-10-01       10
## 4 29.65360 2012-10-01       15
## 5 29.68684 2012-10-01       20
## 6 29.72009 2012-10-01       25
```

```
names(activity_imp)
```

```
## [1] "steps"    "date"     "interval"
```

```
# compare raw and imputed data
# https://www.statmethods.net/stats/descriptives.html
summary(activity)
```

```
##      steps            date              interval
##  Min.   :  0.00   Length:17568       Min.   :   0.0
##  1st Qu.:  0.00   Class :character   1st Qu.: 588.8
##  Median :  0.00   Mode  :character   Median :1177.5
##  Mean   : 37.38                      Mean   :1177.5
##  3rd Qu.: 12.00                      3rd Qu.:1766.2
##  Max.   :806.00                      Max.   :2355.0
##  NA's   :2304
```

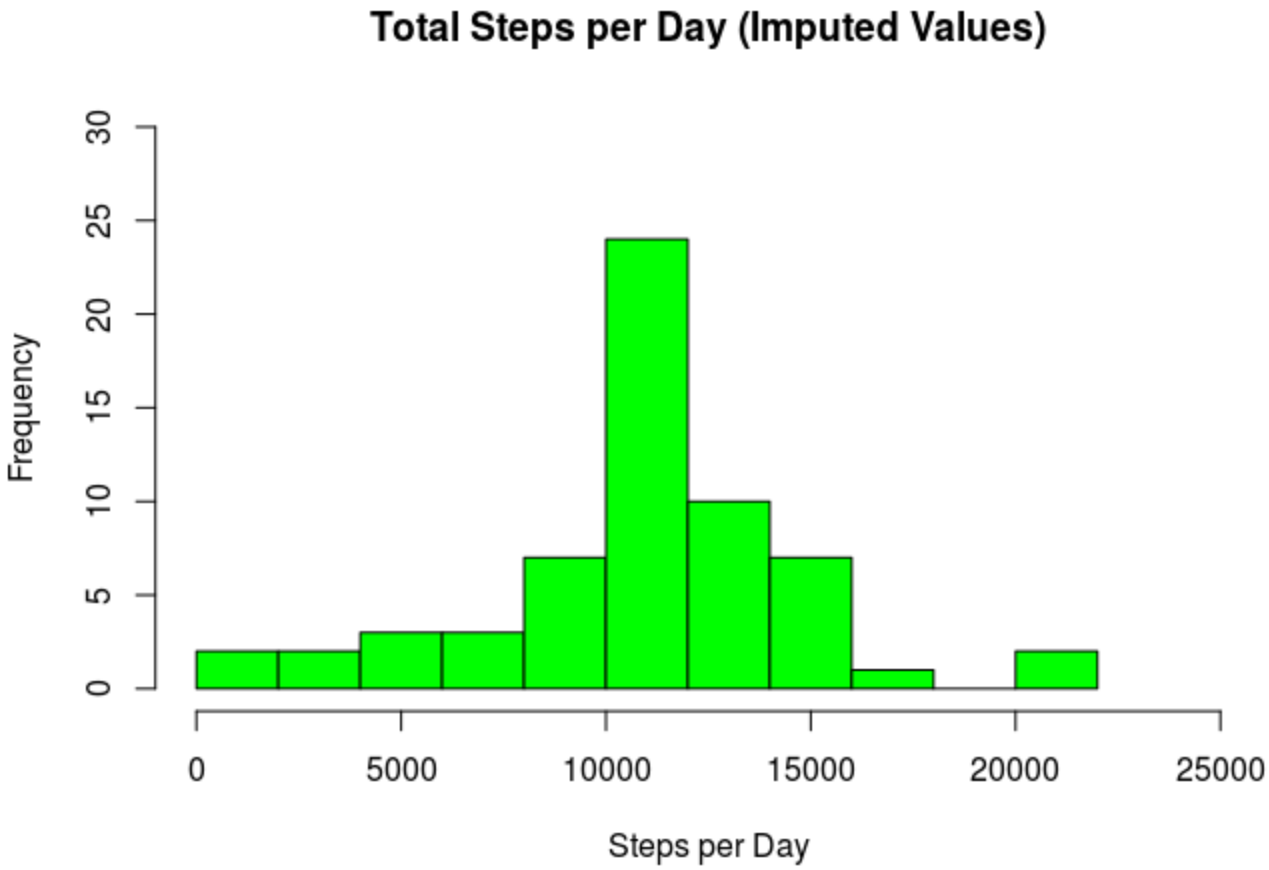
```
summary(activity_imp)
```

```
##      steps            date              interval
##  Min.   :  0.00   Length:17568       Min.   :   0.0
##  1st Qu.:  0.00   Class :character   1st Qu.: 588.8
##  Median :  0.00   Mode  :character   Median :1177.5
##  Mean   : 37.38                      Mean   :1177.5
##  3rd Qu.: 34.37                      3rd Qu.:1766.2
##  Max.   :806.00                      Max.   :2355.0
```

```
# part 4: calculate sum, then create histogram
# aggregate by date, then apply calculations
# source: https://www.statmethods.net/management/aggregate.html
steps_sum_imp <- aggregate(steps ~ date, activity_imp, sum)
summary(steps_sum_imp)
```

```
##      date               steps
##  Length:61          Min.   :   41
##  Class :character   1st Qu.: 9819
##  Mode  :character   Median :10766
##                     Mean   :10766
##                     3rd Qu.:12811
##                     Max.   :21194
```

```
# create histogram plot
# source: https://www.statmethods.net/graphs/density.html
hist(steps_sum_imp$steps, col="green",
     breaks=10, main="Total Steps per Day (Imputed Values)",
     xlab="Steps per Day", ylim=c(0,30), xlim=c(0,25000))
```




```
# aggregate by date, then apply calculations
# source: https://www.statmethods.net/management/aggregate.html
steps_mean_imp <- aggregate(steps ~ date, activity_imp, mean)
summary(steps_mean)
```

```
##      date               steps
##  Length:53          Min.   : 0.1424
##  Class :character   1st Qu.:30.6979
##  Mode  :character   Median :37.3785
##                     Mean   :37.3826
##                     3rd Qu.:46.1597
##                     Max.   :73.5903
```

```
summary(steps_mean_imp)
```

```
##      date              steps
##  Length:61          Min.   : 0.1424
##  Class :character   1st Qu.:34.0938
##  Mode  :character   Median :37.3826
##                     Mean   :37.3826
##                     3rd Qu.:44.4826
##                     Max.   :73.5903
```

```
# aggregate by date, then apply calculations
# source: https://www.statmethods.net/management/aggregate.html
steps_median_imp <- aggregate(steps ~ date, activity_imp, median)
summary(steps_median)
```

```
##      date              steps
##  Length:53          Min.   :0
##  Class :character   1st Qu.:0
##  Mode  :character   Median :0
##                     Mean   :0
##                     3rd Qu.:0
##                     Max.   :0
```

```
summary(steps_median_imp)
```

```
##      date              steps
##  Length:61          Min.   : 0.000
##  Class :character   1st Qu.: 0.000
##  Mode  :character   Median : 0.000
##                     Mean   : 4.903
##                     3rd Qu.: 0.000
##                     Max.   :37.383
```

Are there differences in activity patterns between weekdays and weekends?

Processing Steps: * Created weekday/weekend factor with ifelse logic * Used weekdays function to create weekday/weekend values

Analysis: * Used head, names and summary functions to review the data * Verified that factors were created successfully

Part 2: Create time series facet plot

Processing Steps: * Used dplyr to group by date, then calculate mean

Analysis: * Used head, names and summary functions to review the data * Verified that mean was calculated by date

Plot: * Created time series facet plot; add line color

Notes: * Assumed instructions were for steps/interval across ALL DAYS * So, mean was calculated per interval for entire data set * NOT group by interval, then calculate mean by date

```
# part 1: create weekday/weekend factor within dataset

# first, create weekday/weekend categorical variable
# source: https://sphweb.bumc.bu.edu/otlt/MPH-Modules/BS/R/R-Manual/R-Manual4.html
activity_imp$wday_type <- ifelse(
    weekdays(as.Date(activity_imp$date)) == "Saturday" |
    weekdays(as.Date(activity_imp$date)) == "Sunday",
    "Weekend",
    "Weekday"
)

# verify results
# https://www.statmethods.net/stats/descriptives.html
nrow(activity_imp)
```

```
## [1] 17568
```

```
head(activity_imp)
```

```
##      steps       date interval wday_type
## 1 29.55387 2012-10-01        0   Weekday
## 2 29.58711 2012-10-01        5   Weekday
## 3 29.62036 2012-10-01       10   Weekday
## 4 29.65360 2012-10-01       15   Weekday
## 5 29.68684 2012-10-01       20   Weekday
## 6 29.72009 2012-10-01       25   Weekday
```

```
names(activity_imp)
```

```
## [1] "steps"     "date"      "interval"  "wday_type"
```

```
summary(activity_imp)
```

```
##      steps            date              interval       wday_type
##  Min.   :  0.00   Length:17568       Min.   :   0.0   Length:17568
##  1st Qu.:  0.00   Class :character   1st Qu.: 588.8   Class :character
##  Median :  0.00   Mode  :character   Median :1177.5   Mode  :character
##  Mean   : 37.38                      Mean   :1177.5
##  3rd Qu.: 34.37                      3rd Qu.:1766.2
##  Max.   :806.00                      Max.   :2355.0
```

```
unique(activity_imp$wday_type)
```

```
## [1] "Weekday" "Weekend"
```

```
# group by interval, then calculate mean steps for each group
# source: https://datacarpentry.org/R-genomics/04-dplyr.html
# install.packages("dplyr")
# library(dplyr)
steps_avg_wday <- activity_imp %>%
    group_by(steps, interval, wday_type) %>%
    summarize(steps_avg = mean(steps))
```

```
## `summarise()` regrouping output by 'steps', 'interval' (override with `.groups` argument)
```

```
# verify results
# https://www.statmethods.net/stats/descriptives.html
head(steps_avg_wday)
```

```
## # A tibble: 6 x 4
## # Groups:   steps, interval [3]
##   steps interval wday_type steps_avg
##   <dbl>    <int> <chr>         <dbl>
## 1     0        0 Weekday           0
## 2     0        0 Weekend           0
## 3     0        5 Weekday           0
## 4     0        5 Weekend           0
## 5     0       10 Weekday           0
## 6     0       10 Weekend           0
```

```
names(steps_avg_wday)
```

```
## [1] "steps"     "interval"  "wday_type" "steps_avg"
```

```
summary(steps_avg_wday)
```

```
##      steps           interval     wday_type           steps_avg
##  Min.   :  0.00   Min.   :   0   Length:5235        Min.   :  0.00
##  1st Qu.: 19.00   1st Qu.: 830   Class :character   1st Qu.: 19.00
##  Median : 41.62   Median :1305   Mode  :character   Median : 41.62
##  Mean   :111.60   Mean   :1290                      Mean   :111.60
##  3rd Qu.:107.50   3rd Qu.:1800                      3rd Qu.:107.50
##  Max.   :806.00   Max.   :2355                      Max.   :806.00
```

```
# part 2: create time series facet plot
# source: http://zevross.com/blog/2019/04/02/easy-multi-panel-plots-in-r-using-facet_wrap-and-facet_grid-from-ggplot2/
install.packages("ggplot2")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/4.0'
## (as 'lib' is unspecified)
```

```
library(ggplot2)
ggplot(data=steps_avg_wday, aes(interval, steps_avg)) +
    geom_line(color="blue") +
    geom_point(color="blue") +
    facet_wrap(~ wday_type)
```