Reproducible Research: Peer Assessment 1

Brent Hagel

2024-12-10

Code to load relevant packages for reading data and analysis:

```
library(pacman)

## Warning: package 'pacman' was built under R version 4.4.2

pacman::p_load(
  rio,          # importing data
  here,         # relative file pathways
  janitor,      # data cleaning and tables
  lubridate,    # working with dates
  matchmaker,   # dictionary-based cleaning
  epikit,       # age_categories() function
  tidyverse,    # data management and visualization
  tsibble,      # handle time series datasets
  slider,       # for calculating moving averages
  gridExtra,    #for panel plots with ggplot
  tinytex
)
```

Loading and preprocessing the data

```
download.file("https://github.com/BHagel123/RepData_PeerAssessment1/blob/master/activity.zip?raw=TRUE",
  "activity.zip",
  mode = "wb"
)
unzip("activity.zip")
activity <- import("activity.csv")
skimr::skim(activity) #overview of dataset
```

Data summary

Name	activity
Number of rows	17568
Number of columns	3

Column type frequency:	
Date	1
numeric	2
Group variables	None

Variable type: Date

skim_variable	n_missing	complete_rate	min	max	median	n_unique
date	0	1	2012-10-01	2012-11-30	2012-10-31	61

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
steps	2304	0.87	37.38	112.00	0	0.00	0.0	12.00	806	▇▁▁▁▁
interval	0	1.00	1177.50	692.45	0	588.75	1177.5	1766.25	2355	▇▇▇▇▇

```
#head(activity, 10) #look at first 10 rows
#str(activity) #structure of variables
#summary(activity) #summarize variables in dataset
```

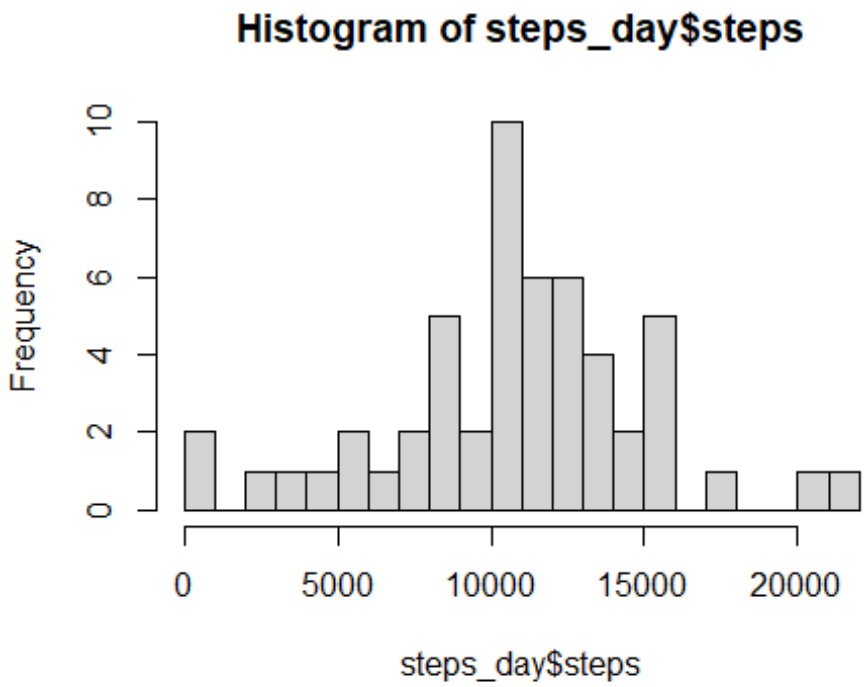
Mean and median number of steps taken each day

1. Calculate the total number of steps taken per day

```
steps_day<-aggregate(steps ~ date, data = activity, FUN = sum) #sum of steps grouped by day
```

2. Histogram of the total number of steps taken each day

```
hist(steps_day$steps, breaks=20)
```



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

```
mean_steps<-mean(steps_day$steps,na.rm=TRUE)
median_steps<-median(steps_day$steps,na.rm=TRUE)
```

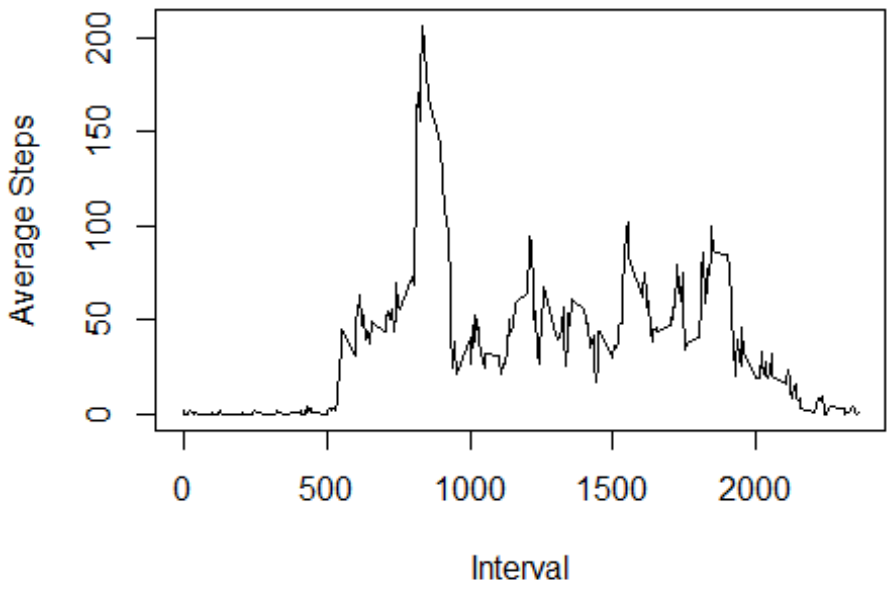
The mean steps per day = 1.0766189^{4}.

The median steps per day = 10765.

Time series plot of the average number of steps taken

1. Make a time series plot (i.e. type = “l”) of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all days (y-axis)

```
activity_grp<-aggregate(steps ~ interval, data = activity, FUN = mean) #mean steps grouped by interval
#head(activity_grp$steps)
plot(activity_grp$interval, activity_grp$steps, type="l", ylab="Average Steps", xlab="Interval")
```




```
#activity_grp <- tsibble(activity_grp, index = interval) #alternative
plotting approach with ggplot
#ggplot(activity_grp, aes(x = interval, y = steps)) +
#     geom_line()
```

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

```
max_interval<-activity_grp %>%
  filter(activity_grp$steps==max(activity_grp$steps)) %>% # keep the row with
the max steps
select(interval) #report back the interval
```

The interval with the maximum number of steps is 835.

Imputing missing values

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

```
activity_NA<-activity %>%
filter(is.na(steps)) %>% #filter dataset to number of rows where steps == NA
select(steps)
nrow(activity_NA)
```

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

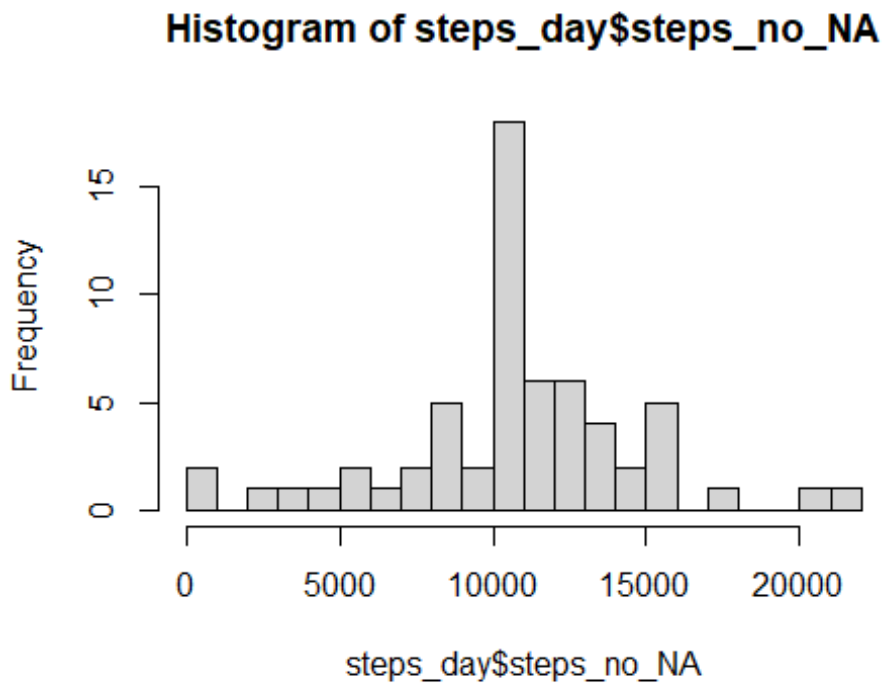
The total number of missing values in the dataset: 2304

2. 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. 3. Create a new dataset that is equal to the original dataset but with the missing data filled in.

```
activity_grp<-aggregate(steps ~ interval, data = activity, FUN = mean) #mean
steps grouped by date
#str(activity)
#str(activity_grp)
activity_merge<-merge(activity, activity_grp, by.x="interval",
by.y="interval", all=TRUE) #merge together two datasets
#str(activity_merge)
activity_merge<-activity_merge %>%
mutate(steps_no_NA=if_else(is.na(steps.x),steps.y, steps.x))
#str(activity_merge)
```

4. 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. 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?

```
steps_day<-aggregate(steps_no_NA ~ date, data = activity_merge, FUN = sum)
#mean steps grouped by day
hist(steps_day$steps_no_NA, breaks=20)
```




```
mean(steps_day$steps_no_NA)

## [1] 10766.19

median(steps_day$steps_no_NA)

## [1] 10766.19
```

The number of steps per day essentially remained unchanged after imputing missing values with the mean of the interval.

Are there differences in activity patterns between weekdays and weekends?

1. 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.

```
activity_merge<-activity_merge %>%
mutate(weekday=weekdays(activity_merge$date))
#str(activity_merge)
activity_merge$weekend[activity_merge$weekday == "Saturday" |
activity_merge$weekday == "Sunday"] <- "Weekend"
activity_merge$weekend[is.na(activity_merge$weekend) == TRUE] <- "Weekday"
```

```
activity_merge$weekend <- factor(activity_merge$weekend, levels=c("Weekend",
"Weekday"), labels=c("Weekend", "Weekday"))
#str(activity_merge) #used to check structure of dataframe
#head(activity_merge,20) #used to see dataframe values
```

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

```
activity_grp1<-aggregate(steps_no_NA ~ interval+weekend, data =
activity_merge, FUN = mean) #mean steps grouped by interval and weekend
#str(activity_grp1)
activity_grp_we<- activity_grp1 %>% filter(weekend == "Weekend")
activity_grp_wd<- activity_grp1 %>% filter(weekend == "Weekday")
weekend_plot<-ggplot(activity_grp_we, aes(x = interval, y = steps_no_NA)) +
geom_line() +
  labs(title = "Weekend", y="Average number of steps", x="Interval") +
ylim(0,250)
weekday_plot<-ggplot(activity_grp_wd, aes(x = interval, y = steps_no_NA)) +
geom_line() +
  labs(title = "Weekday", y="Average number of steps", x="Interval") +
ylim(0,250)
grid.arrange(weekend_plot, weekday_plot, nrow = 2)
```