RR Course Project 1

KShear

8/6/2020

Data Prep

This project starts by loading the provded activity data, checking the structure of the dataframe and then aggregates the step variable by date in preparation for the next step.

```
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
```

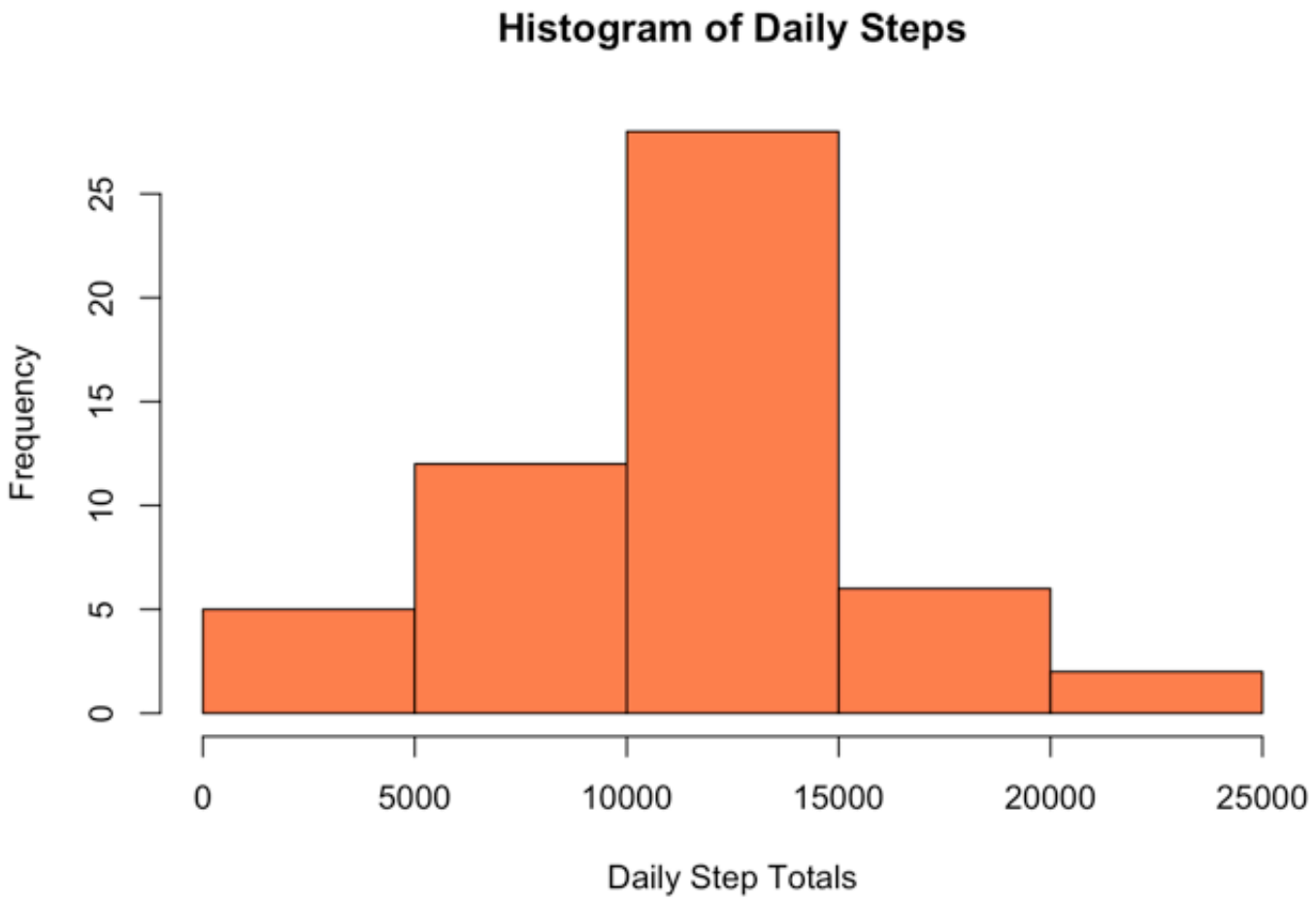
```
library(lattice)

#read in data file
activity <-read.csv("activity.csv")
#view str of data
str(activity)
```

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

```
#create variable for daily totals
daily<- aggregate(steps ~ date, activity, sum)
```

Histogram of the total number of steps taken each day

```
#1. create histogram of daily step count
hist(daily$steps, xlab="Daily Step Totals", main= "Histogram of Daily Steps", col= "c
oral")
```



Mean and median number of steps taken each day

```
#2. calculate mean
mean(daily$steps)
```

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

```
#3. calculate median
median(daily$steps)
```

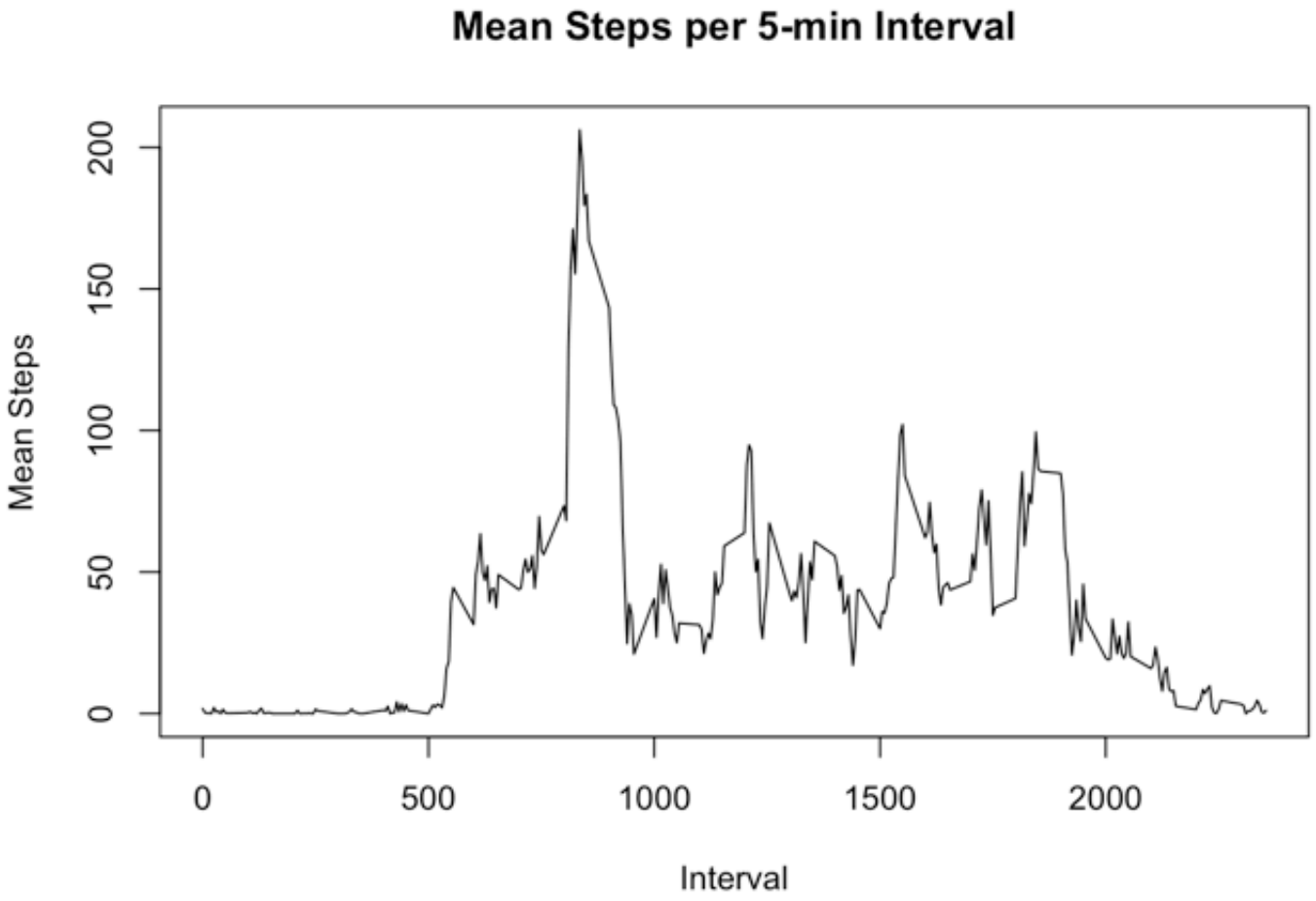
```
## [1] 10765
```

##Time series plot of the average number of steps taken based on raw data

```
#4. time series plot of average number of steps taken
    #calculate mean for intervals to generate plot
IntAvg <-  aggregate(steps ~ interval, activity, mean, is.na=F) #change made here 12:
29)
    #check structure of new dataframe
str(IntAvg)
```

```
## 'data.frame':    288 obs. of  2 variables:
##  $ interval: int  0 5 10 15 20 25 30 35 40 45 ...
##  $ steps   : num  1.717 0.3396 0.1321 0.1509 0.0755 ...
```

```
    #create plot using base
DailyPat <- plot(x=IntAvg$interval, y=IntAvg$steps, type='l', xlab="Interval", ylab="
Mean Steps", main="Mean Steps per 5-min Interval")
```



##The 5-minute interval that, on average, contains the maximum number of steps

```
#5. The 5-minute interval that, on average, contains the maximum number of steps

summary(IntAvg)
```

```
##     interval          steps
##  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
```

```
#View(IntAvg) #can view to see what the largest number is
max <- filter(IntAvg, steps >= 206) #set to filter to only the interval with the high
est step avg
print(max$interval) #prints the interval number with the highest step avg
```

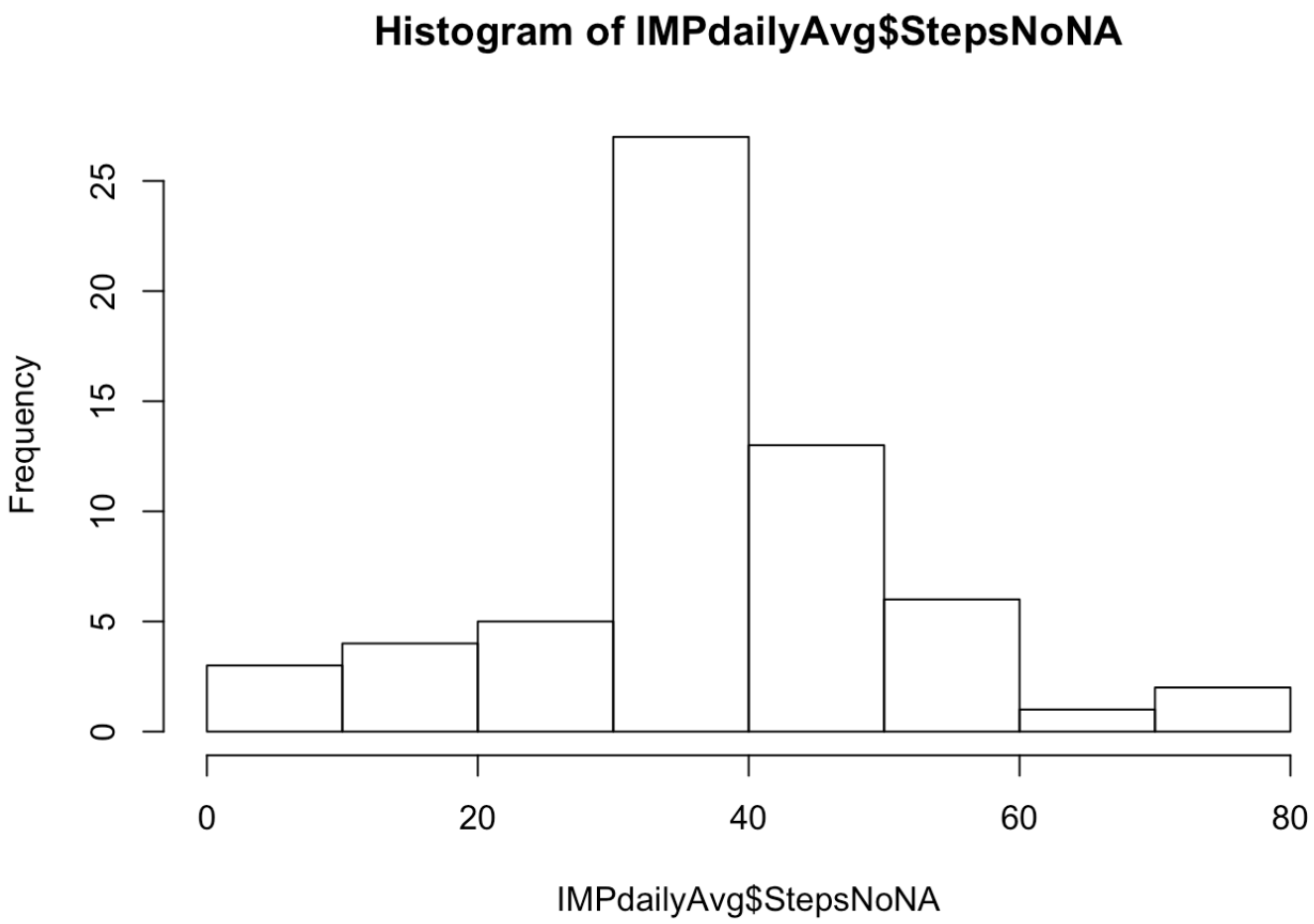
```
## [1] 835
```

##Code to describe and show a strategy for imputing missing data

```
#6.Code to describe and show a strategy for imputing missing data

    #Find mean steps per interval so we know what number to replace the NAs with medi
an=0 so mean will be used

    StepPerInt <- activity$steps #copy this row so that we can then impute into this
vector

    IntAvg <-  aggregate(steps ~ interval, activity, mean)#aggregate the mean steps f
or each interval
    MeanIntAvg <- mean(IntAvg$steps) #calculates the mean across all intervals (this
is the number to impute)

    #Find number of missing values (does this change all the NAs to 0's??)
    #NumNA <- sum(is.na(activity$steps))
    #print(NumNA)

  StepPerInt[which(is.na(StepPerInt))] <- 37.38 #this imputese the mean steps per int
erval for all NA values

  StepsNoNA <- StepPerInt #applies a more desciptive label

  activityIMP <- data.frame(activity, StepsNoNA) #creates new dataframe that includes
the StepsNonNA data
```

Histogram of the total number of steps taken each day after missing values are imputed

```
#7. Histogram of the total number of steps taken each day after missing values are im
puted
    IMPdailyAvg <-  aggregate(StepsNoNA ~ date, activityIMP, mean)#aggregate the mean
steps for each day

    hist(IMPdailyAvg$StepsNoNA) #generates the histogram for imputed data
```



Panel plot comparing the average number of steps taken per 5-minute interval across weekdays and weekends

```
#8. Panel plot comparing the average number of steps taken per 5-minute interval acro
ss weekdays and weekends

    library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

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

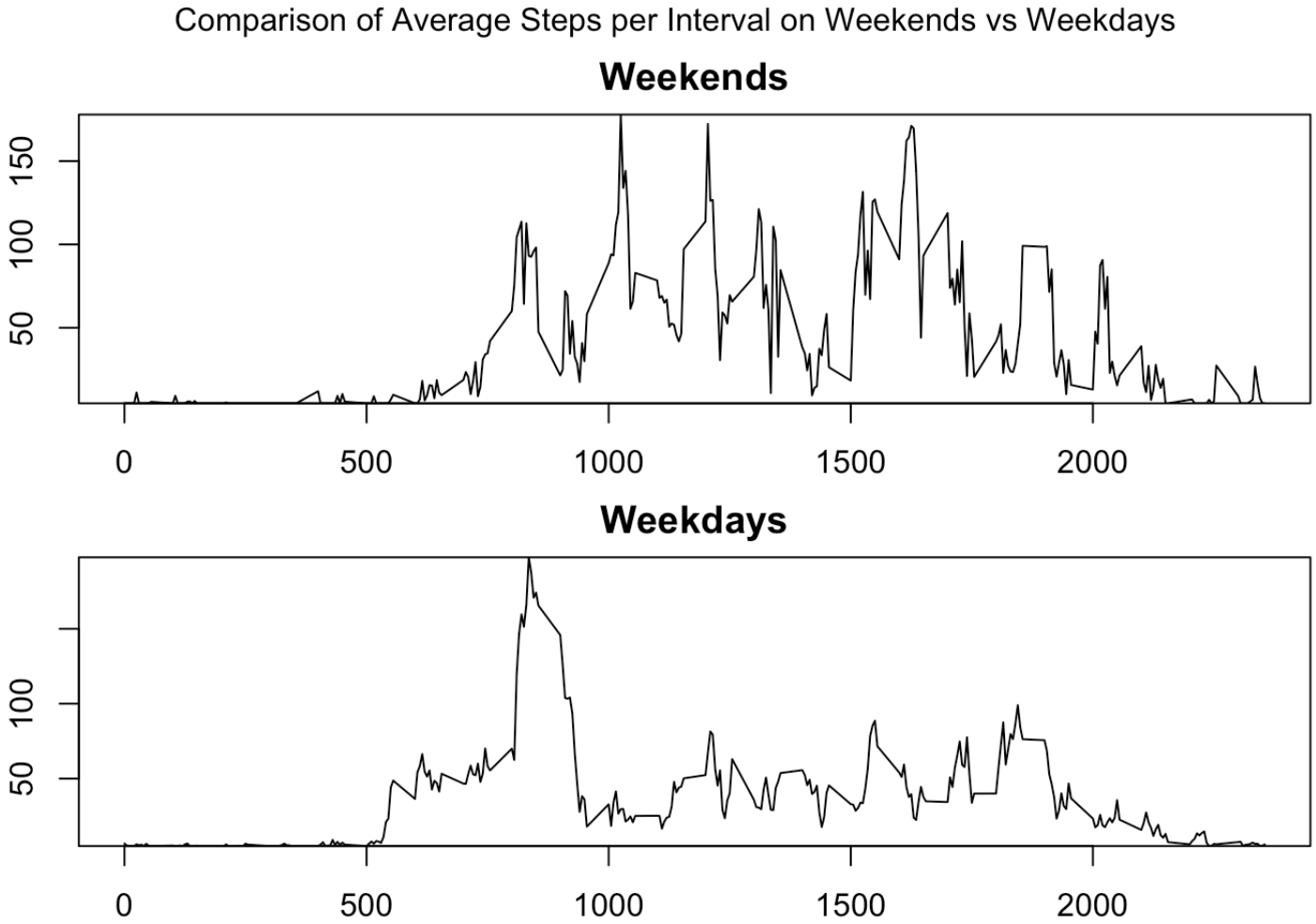
```
    dow <- wday(activityIMP$date) ##genrates new value where number 1-7 represents Su
nday - Saturday (weekends = 1 & 7)
```

```
## Warning: tz(): Don't know how to compute timezone for object of class factor;
## returning "UTC". This warning will become an error in the next major version of
## lubridate.
```

```
    weekend <- dow == (1 | 7) #generates a logical vector where "TRUE" is for weekend days
    summary(weekend)
```

```
##    Mode   FALSE    TRUE
## logical   15264    2304
```

```
    activityIMPdow <- data.frame(activityIMP, dow, weekend) #creates dataframe that includes dow and weekend variables

    StepsWeekend <- filter(activityIMPdow, weekend == TRUE) #filters data to just weekend days
    StepsWeek <- filter(activityIMPdow, weekend == FALSE) #filters data to just week days

    AvgWeekend <-  aggregate(StepsNoNA ~ interval, StepsWeekend, mean) #Calculates Avg steps per interval for weekends
  #View(AvgWeekend)
    AvgWeek <- aggregate(StepsNoNA ~ interval, StepsWeek, mean) #Calculates Avg steps per interval for weekdays
 #generates plots
    par(mfcol = c(2,1), mar = c(2, 2, 2, 1), oma = c(0, 0, 2, 0)) #bottom, left, top, right
    plot(AvgWeekend$interval, AvgWeekend$StepsNoNA, type = 'l', yaxs= "i", xlab="interval", ylab="# of steps", main="Weekends")

    plot(AvgWeek$interval, AvgWeek$StepsNoNA, type = 'l', yaxs = "i", xlab="interval", ylab="# of steps", main="Weekdays")

    mtext("Comparison of Average Steps per Interval on Weekends vs Weekdays", outer = T)
```


Comparison of Average Steps per Interval on Weekends vs Weekdays
Weekends
50
100
150
0
500
1000
1500
2000
Weekdays
50
100
0
500
1000
1500
2000