week2 project

zly

2021/7/11

```
data <- read.csv("activity.csv",header = T,sep = ",")
head(data)
```

```
##   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
```

```
## 1.Calculate the total number of steps taken per day
library(dplyr)
```

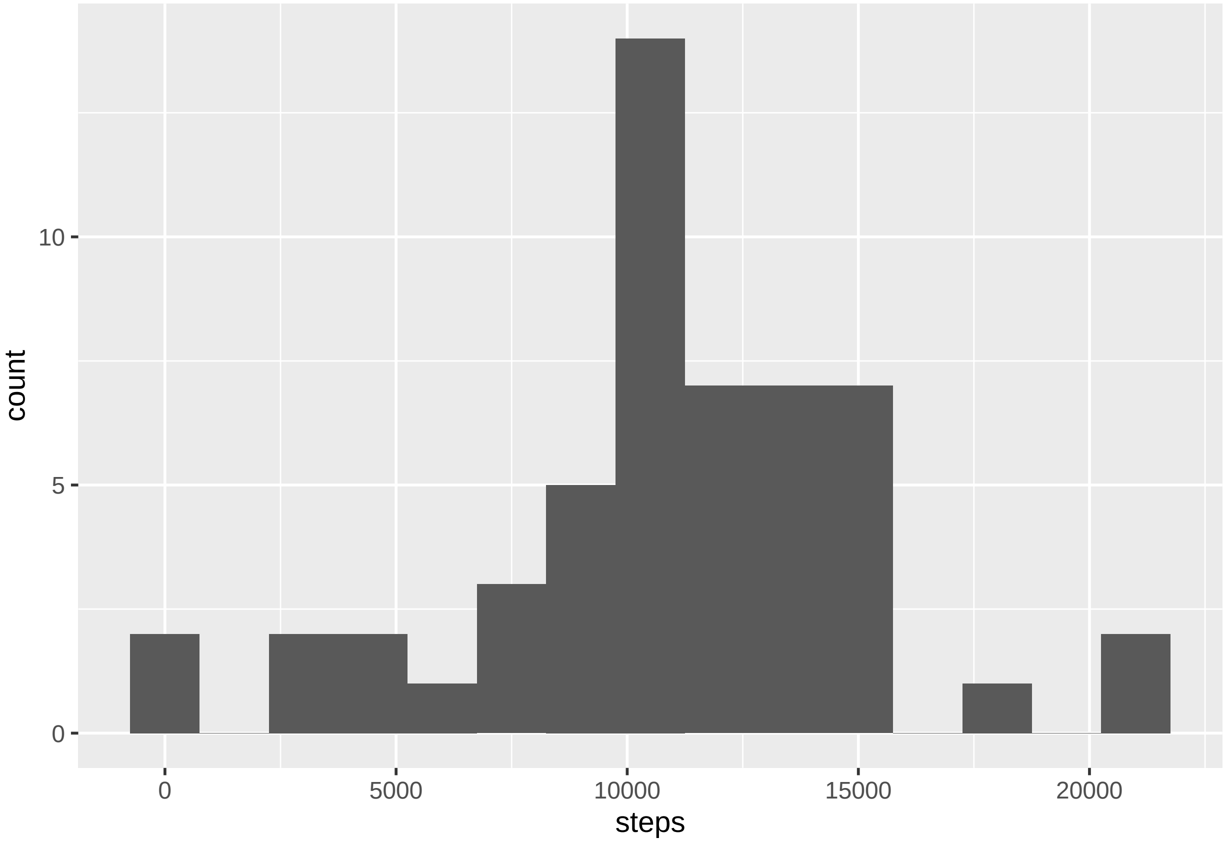
```
##
## Attaching package: 'dplyr'
```

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

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

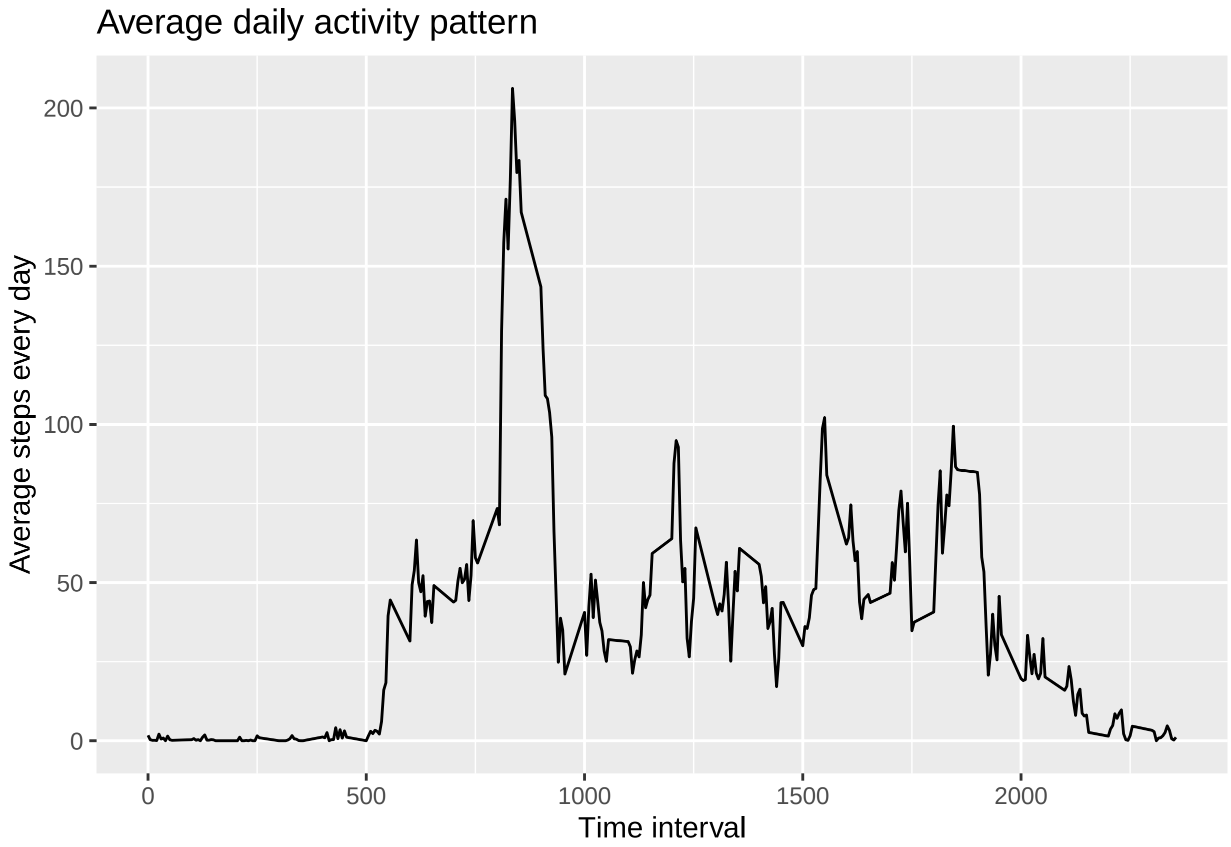
```
everyday_step <- aggregate(data$steps, by=list(type=data$date),sum)
colnames(everyday_step) <- c("date","steps")
## 2. Create a histogram
library(ggplot2)
ggplot(everyday_step, aes(steps)) +geom_histogram(binwidth = 1500)
```

```
## Warning: Removed 8 rows containing non-finite values (stat_bin).
```



```
## 3.Calculate and report the mean and median of the total number of steps taken per day
meanstep <- mean(everyday_step$steps,na.rm = T)
medianstep <- median(everyday_step$steps,na.rm = T)
```

```
## 1.Make a time series plot (i.e. \color{red}{\verb|type = "l"|}type = "l") of the 5-minute interval (
averages <- aggregate(x=list(steps=data$steps), by=list(interval=data$interval),
                      FUN=mean, na.rm=TRUE)
ggplot(averages, aes(interval, steps)) + geom_line(color = "black", size =0.5) +
  labs( y = "Average steps every day",
        x = "Time interval",
        title = "Average daily activity pattern")
```



```
## 2.Which 5-minute interval, on average across all the days in the dataset, contains the maximum numbe
highest <- averages[which.max(averages$steps),]
highest
```

```
##     interval    steps
## 104      835 206.1698
```

```
## 1.Calculate and report the total number of missing values in the dataset (i.e. the total number of r
sapply(X = data, FUN = function(x) sum(is.na(x)))
```

```
##    steps     date interval
##     2304        0        0
```

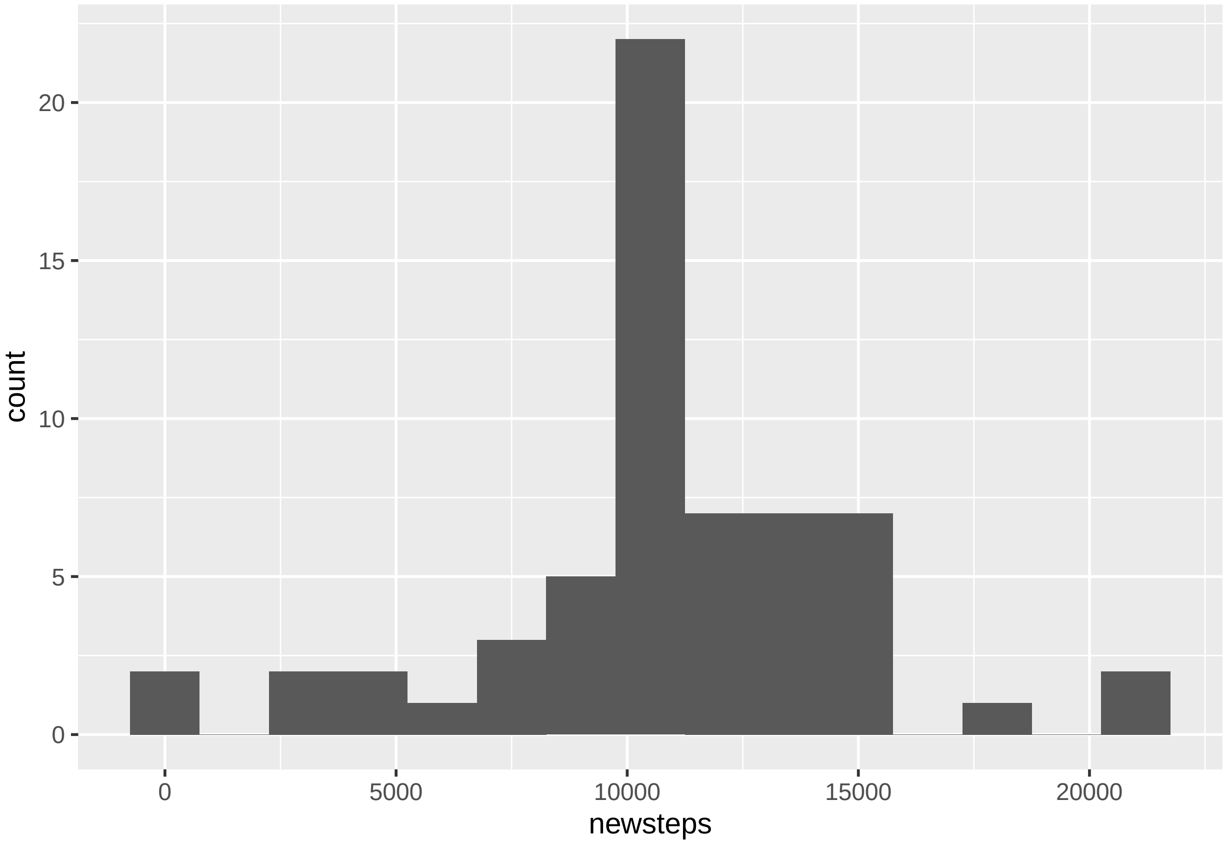
```
## 2.Devise a strategy for filling in all of the missing values in the dataset. The strategy does not n
replaceNA <- function(num)
{replace(num, is.na(num), mean(num, na.rm = TRUE))
  }
meanday <- (data %>% group_by(interval) %>% mutate(steps = replaceNA(steps)))
head(meanday)
```

```
## # A tibble: 6 x 3
## # Groups:   interval [6]
##    steps date       interval
##    <dbl> <chr>         <int>
## 1 1.72   2012-10-01        0
## 2 0.340  2012-10-01        5
## 3 0.132  2012-10-01       10
## 4 0.151  2012-10-01       15
## 5 0.0755 2012-10-01       20
## 6 2.09   2012-10-01       25
```

```
## 3.Create a new dataset that is equal to the original dataset but with the missing data filled in.
meanday_new <- as.data.frame(meanday)
head(meanday_new)
```

```
##       steps       date interval
## 1 1.7169811 2012-10-01        0
## 2 0.3396226 2012-10-01        5
## 3 0.1320755 2012-10-01       10
## 4 0.1509434 2012-10-01       15
## 5 0.0754717 2012-10-01       20
## 6 2.0943396 2012-10-01       25
```

```
## 4.Make a histogram of the total number of steps taken each day and Calculate and report the mean and
new_everyday_step <- aggregate(meanday_new$steps, by = list(meanday_new$date), sum)
colnames(new_everyday_step) <- c("date","newsteps")
ggplot(new_everyday_step, aes(newsteps)) +geom_histogram(binwidth = 1500)
```



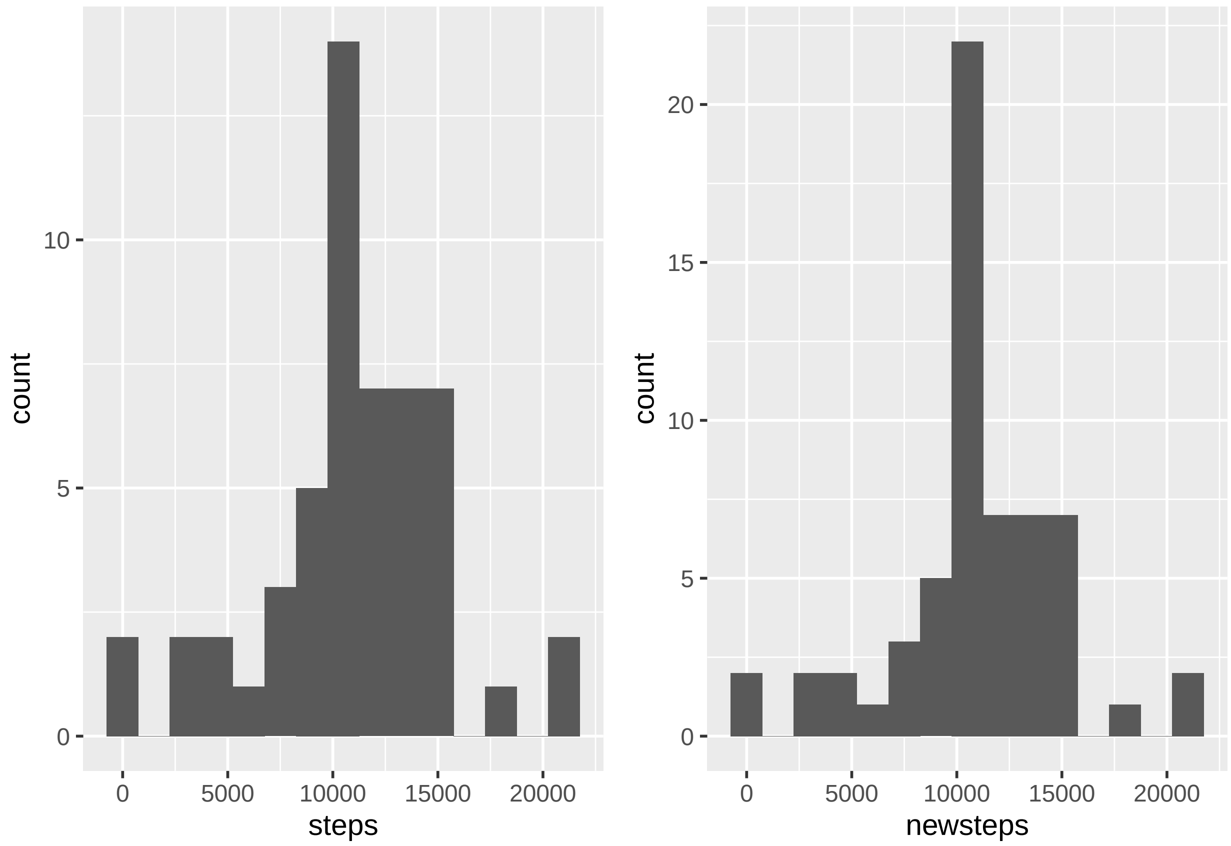
```
library(grid)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
part1 <- ggplot(everyday_step, aes(steps))+geom_histogram(binwidth = 1500)
part2 <- ggplot(new_everyday_step, aes(newsteps))+geom_histogram(binwidth = 1500)
grid.arrange(part1, part2, ncol = 2)
```

```
## Warning: Removed 8 rows containing non-finite values (stat_bin).
```



```r
mean(na.omit(everyday_step$steps))
```

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

```r
median(na.omit(everyday_step$steps))
```

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

```r
mean(new_everyday_step$newsteps)
```

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

```r
median(na.omit(new_everyday_step$newsteps))
```

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

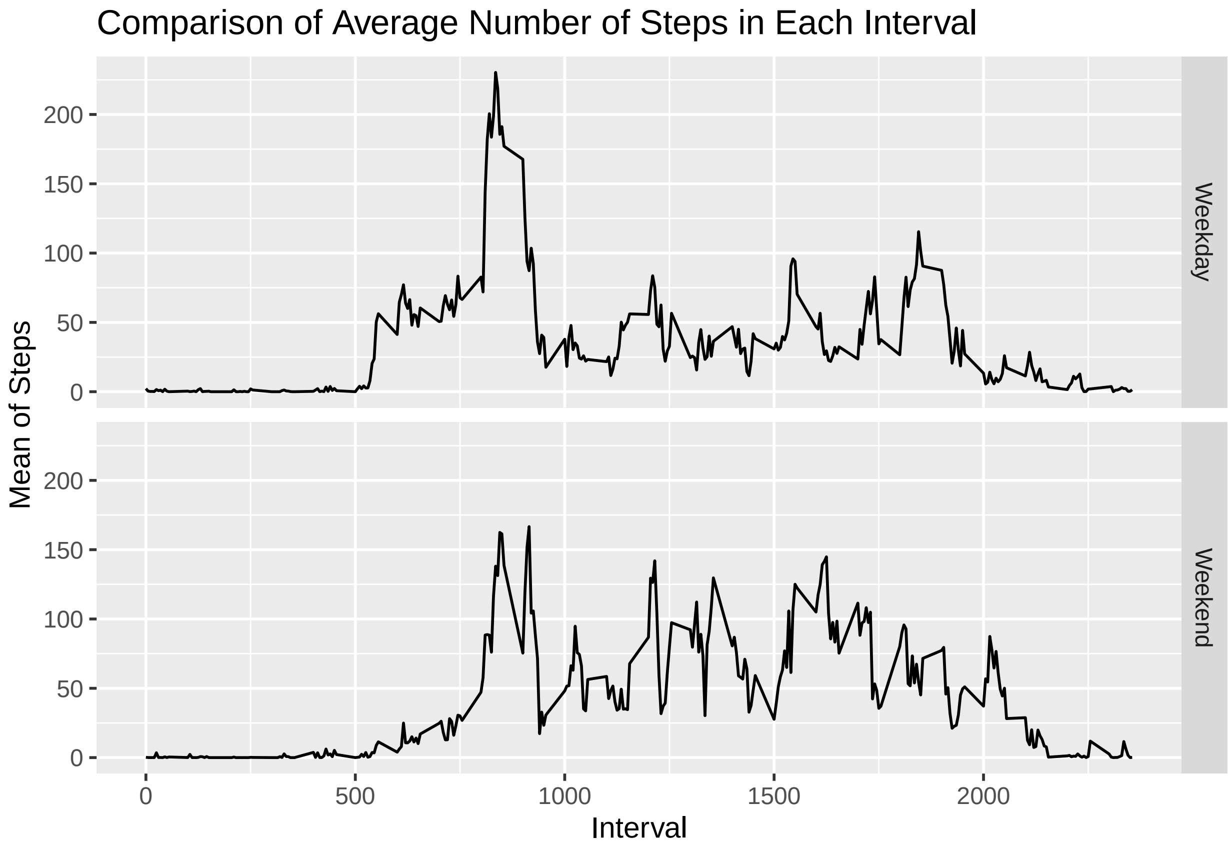
```r
## 1.Create a new factor variable in the dataset with two levels - "weekday" and "weekend" indicating w
meanday_new$date <- as.Date(meanday_new$date, format = "%Y-%m-%d")
weekday <- weekdays(meanday_new$date)
meanday_new <- cbind(meanday_new,weekday)
meanday_new$weekday <- as.character(meanday_new$weekday)
## 2.Make a panel plot containing a time series plot (i.e. \color{red}{\verb|type = "l"|}type = "l") of
meanday_new$group <- ifelse(meanday_new$weekday %in% c("Monday", "Tuesday", "Wednesday", "Thursday", "F
head(meanday_new)
```

```
##       steps       date interval weekday   group
## 1 1.7169811 2012-10-01        0  Monday Weekday
## 2 0.3396226 2012-10-01        5  Monday Weekday
## 3 0.1320755 2012-10-01       10  Monday Weekday
## 4 0.1509434 2012-10-01       15  Monday Weekday
## 5 0.0754717 2012-10-01       20  Monday Weekday
## 6 2.0943396 2012-10-01       25  Monday Weekday
```

```
newdat <- (meanday_new %>% group_by(interval, group) %>% summarise(Mean = mean(steps)))
```

```
## `summarise()` has grouped output by 'interval'. You can override using the `.groups` argument.
```

```
ggplot(newdat, mapping = aes(x = interval, y = Mean)) + geom_line() +
  facet_grid(group ~.) + xlab("Interval") + ylab("Mean of Steps") +
  ggtitle("Comparison of Average Number of Steps in Each Interval")
```


Comparison of Average Number of Steps in Each Interval
Mean of Steps
Weekday
Weekend
Interval