Reproducible Research: Peer Assessment 1

Loading and preprocessing the data

1. Load the data The code below creates a “coursera_reproducible_research” directory within user’s current working directory, downloads the datafile directly from the source, unzips the file and reads it in using the data.table package. The first few entries of the dataset are presented in the output below.

```
library("data.table")

if (!file.exists("coursera_reproducible_research")){dir.create("coursera_reproducible_research"
)}
fileUrl <- "https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip"
download.file(fileUrl, destfile = "./coursera_reproducible_research/repdata%2Fdata%2Factivity.zi
p", method = "curl")
unzip(zipfile="./coursera_reproducible_research/repdata%2Fdata%2Factivity.zip",
    exdir="coursera_reproducible_research")
activity_data<-read.csv("./coursera_reproducible_research/activity.csv", header=TRUE)
head(activity_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
```

2. Process/transform the data (if necessary) into a format suitable for your analysis.

Other than having NAs, in the steps column, the data appears to be clean. Date varaible is stored as a factor so let’s change it to the date format.

```
activity_data$date<-as.Date(as.character(activity_data$date),"%Y-%m-%d")

head(activity_data, n=3)
```

```
##   steps       date interval
## 1    NA 2012-10-01        0
## 2    NA 2012-10-01        5
## 3    NA 2012-10-01       10
```

What is mean total number of steps taken per day?

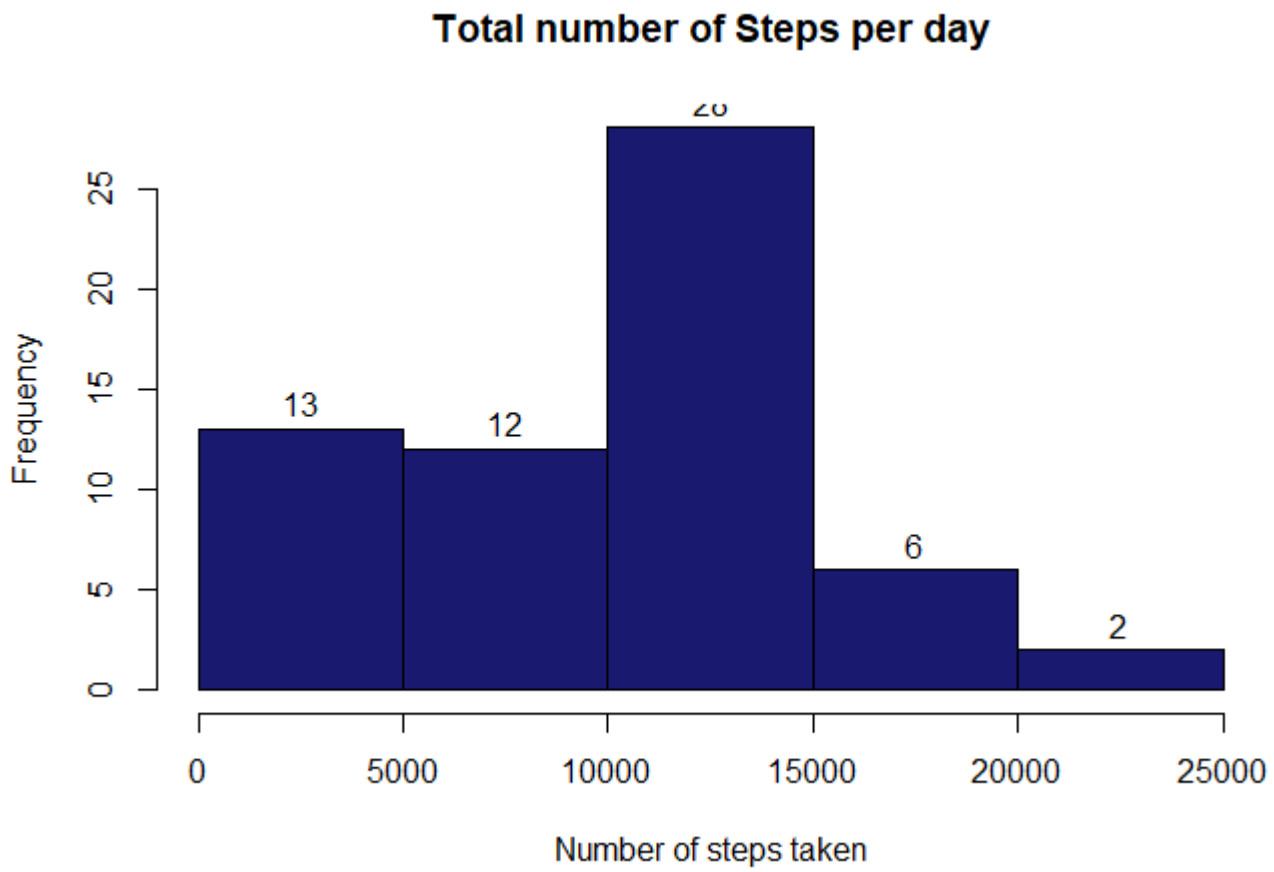
1. Calculate the total number of steps taken per day

```
total_per_day<-with(activity_data, tapply(steps, date, sum, na.rm=TRUE))
head(total_per_day)
```

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

2. Make a histogram of the total number of steps taken each day

```
library(graphics)
for_hist <- hist(total_per_day, plot = FALSE)
plot(for_hist, labels = TRUE, col = "Midnight Blue", main="Total number of Steps per day", xlab=
"Number of steps taken")
```



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

```
mean_per_day<-mean(total_per_day, na.rm=TRUE)
#print(paste("Mean number of steps per day is", round(mean_per_day,2)))
#print(paste("Median number of steps per day is", round(median(total_per_day),2)))
median_per_day<-median(total_per_day, na.rm =TRUE)
```

The mean and median values are

```
#install.packages("xtable")
library(xtable)
```

```
## Warning: package 'xtable' was built under R version 3.5.3
```

```
results_table<- cbind(mean_per_day,median_per_day)
colnames(results_table)<-c("Mean", "Median")
print(results_table, type="html")
```

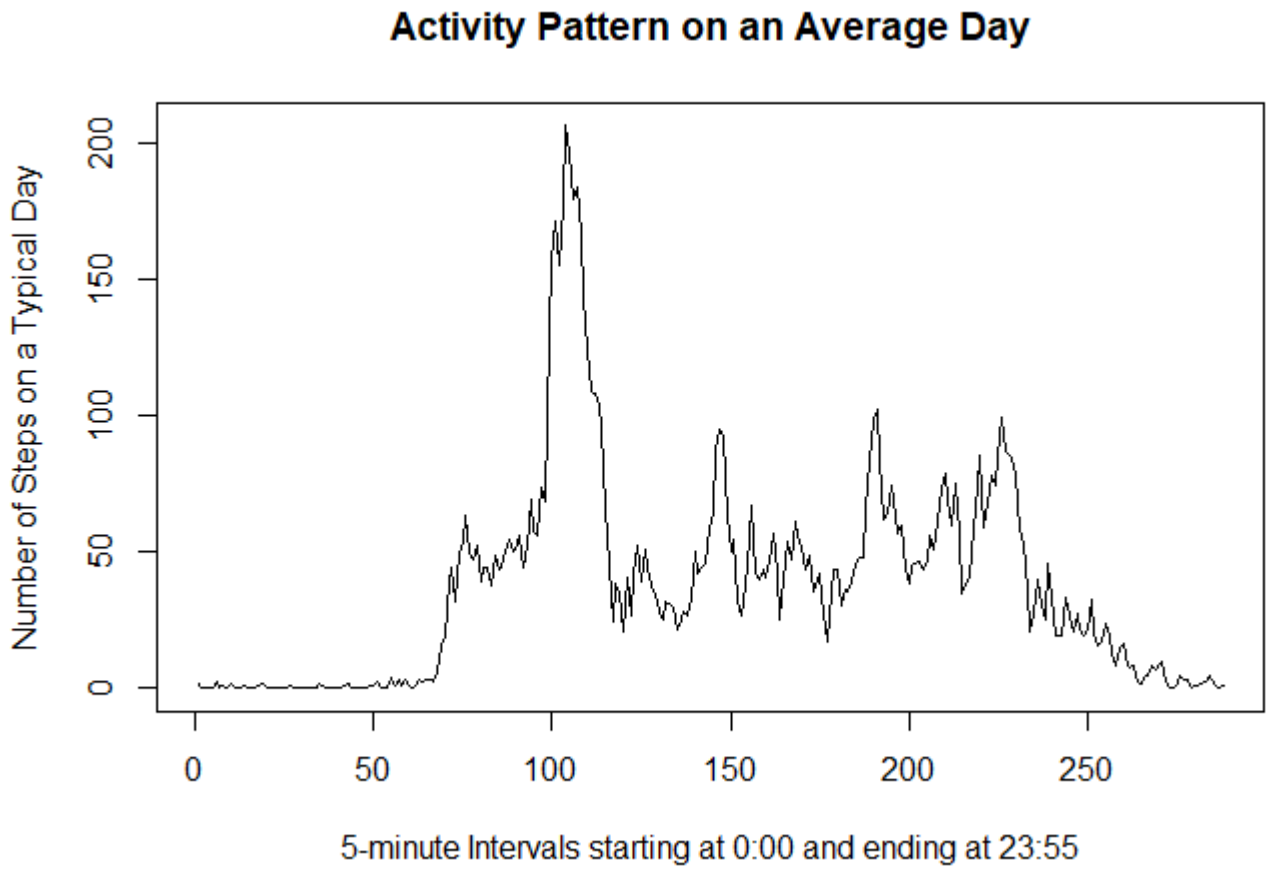
```
    Mean Median
```

[1,] 9354.23 10395

What is the average daily activity pattern?

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)

```
daily_activity_average<-by(activity_data$steps,activity_data$interval,mean, na.rm=TRUE)

plot(daily_activity_average, type="l", main="Activity Pattern on an Average Day", xlab="5-minute
Intervals starting at 0:00 and ending at 23:55", ylab="Number of Steps on a Typical Day")
```



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

```
max_steps<-max(daily_activity_average)
max_step_index<-which(daily_activity_average==max_steps)
max_interval<-names(daily_activity_average[max_step_index])
print(paste("The maximum number of steps is ", round(max_steps,0), " and occurs in the interval
 ", max_interval ))
```

```
## [1] "The maximum number of steps is  206  and occurs in the interval  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)

```
missing_values<-sum(is.na(activity_data))
print(paste("There are ", missing_values, " missing values"))
```

```
## [1] "There are  2304  missing values"
```

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.

```
#subset the original data set so that only rows with misisng values are in the new dataset
activity_data_missing<-activity_data[is.na(activity_data),]
summary(activity_data_missing)
```

```
##      steps           date               interval
##  Min.   : NA    Min.   :2012-10-01   Min.   :   0.0
##  1st Qu.: NA    1st Qu.:2012-10-26   1st Qu.: 588.8
##  Median : NA    Median :2012-11-06   Median :1177.5
##  Mean   :NaN    Mean   :2012-11-01   Mean   :1177.5
##  3rd Qu.: NA    3rd Qu.:2012-11-11   3rd Qu.:1766.2
##  Max.   : NA    Max.   :2012-11-30   Max.   :2355.0
##  NA's   :2304
```

```
# the only misisng data is in the steps column

# General Strategy: extract the interval for each missing values and impute median for that inte
rval

# compute median for each interval
daily_activity_median<-with(activity_data, tapply(steps,interval,median, na.rm=TRUE))

# impute values

for (i in 1: dim(activity_data_missing)[1]){
  # read off the interval for this row
  current_interval<- activity_data_missing$interval[i]
  # get median value for that interval
  median_steps<-daily_activity_median[names(daily_activity_median)%in%     as.character(current_
interval) ]
  activity_data_missing$steps[i]<-median_steps
}
```

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

```
# remove NAs from the original dataset and impute previously-computed values back in

complete_cases_data<-activity_data[complete.cases(activity_data), ]

final_data_set<-rbind(complete_cases_data, activity_data_missing)
head(final_data_set)
```

```
##     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
```

```
# check that dimesions match
dim(activity_data)[1]
```

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

```
dim(complete_cases_data)[1] + dim(activity_data_missing)[1]
```

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

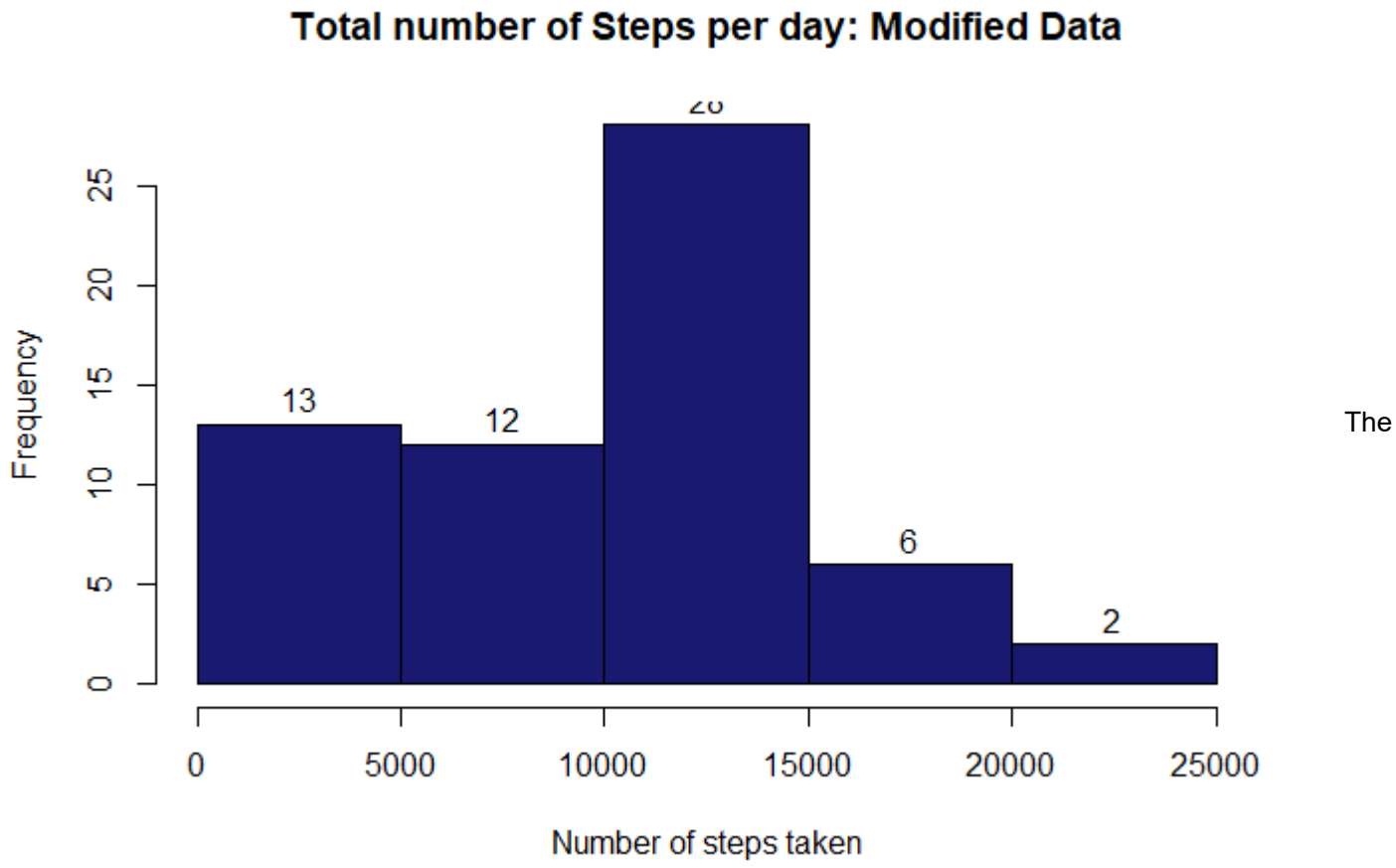
```
dim(final_data_set)[1]
```

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

```
# Everything Matches. Good
```

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?

```
#compute totals per day
total_Steps<-with(final_data_set, tapply(steps, date,sum))
library(graphics)
for_hist <- hist(total_Steps, plot = FALSE)
plot(for_hist, labels = TRUE, col = "Midnight Blue", main="Total number of Steps per day: Modifi
ed Data", xlab="Number of steps taken")
```



The mean and median values are

```
#install.packages("xtable")
library(xtable)
daily_mean<-mean(total_Steps)
daily_median<-median(total_Steps)
mean_and_medain<- cbind(daily_mean,daily_median)
colnames(mean_and_medain)<-c("Mean", "Median")
print(mean_and_medain, type="html")
```

```
     Mean Median
```

[1,] 9503.869 10395

There is a difference in the mean values but the median value has been preserved because I imputed with the median.

```
#install.packages("xtable")
library(xtable)
summary_table<- rbind(results_table,mean_and_medain)
rownames(summary_table)<-c("Original Data", "With Imputed Values")
print(summary_table, type="html")
```

```
                    Mean Median
```

Original Data 9354.230 10395 With Imputed Values 9503.869 10395 ## Are there differences in activity patterns between weekdays and weekends? For this part the weekdays() function may be of some help here. Use the dataset with the filled-in missing values for this part.

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.

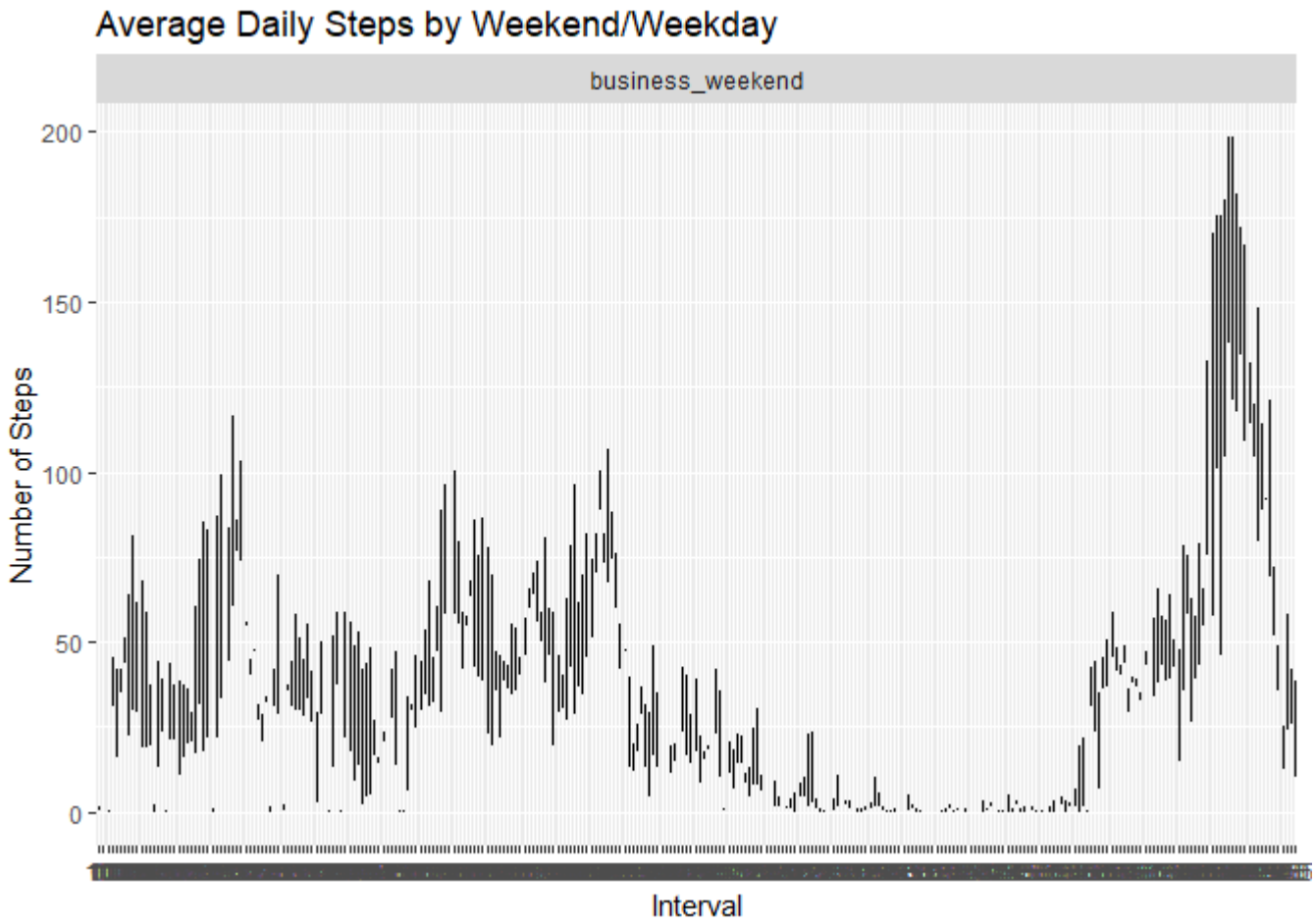
```
final_data_set$date<-as.POSIXct(final_data_set$date, format = "%Y-%m-%d")
final_data_set$dow<- weekdays(final_data_set$date)
weekday<-c("Monday","Tuesday","Wednesday","Thursday","Friday")
final_data_set$business_weekend<-NA
final_data_set[final_data_set$dow%in%weekday,"business_weekend"]<-"weekday"
final_data_set[is.na(final_data_set$business_weekend),"business_weekend"]<-"weekend"
final_data_set$business_weekend<-as.factor(final_data_set$business_weekend)
```

2. 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). See the README file in the GitHub repository to see an example of what this plot should look like using simulated data.

```
weekend_data<-final_data_set[final_data_set$business_weekend=="weekend",]
weekday_data<-final_data_set[final_data_set$business_weekend=="weekday",]

weekend_averages<-with(weekend_data, tapply(steps, interval, mean))

weekday_averages<-with(weekday_data, tapply(steps, interval, mean))


data<-rbind(weekend_averages,weekday_averages)
data2<-t(data)
data2<-as.data.frame(unlist(data2))
data2$interval<-row.names(data2)
reshaped_data<-melt(data2, id.vars=c("interval"),measure.vars=c("weekend_averages", "weekday_ave
rages"), variable.name="business_weekend",value.name="steps")
head(reshaped_data)
```

```
##   interval business_weekend     steps
## 1        0 weekend_averages 0.5882353
## 2        5 weekend_averages 0.0000000
## 3       10 weekend_averages 0.0000000
## 4       15 weekend_averages 0.0000000
## 5       20 weekend_averages 0.0000000
## 6       25 weekend_averages 5.1176471
```

```
library(ggplot2)

ggplot(reshaped_data , aes(interval, steps)) + facet_wrap(vars("business_weekend"))+geom_line()
 + labs(title = "Average Daily Steps by Weekend/Weekday", x = "Interval", y = "Number of Steps")
```


Average Daily Steps by Weekend/Weekday
business_weekend
200
150
100
50
0
Number of Steps
Interval