Intro to Heart Rate

Corey Kimzey

2017-10-18

Description

This package is used to do analysis on some resting heart rate data I have been collecting. It creates target heart ranges for excersing. Also it compares heart rate to it's moving average. The package also tries to fit some linear models. It can also create a scatter plot with the correlation listed as a title.

```
library(knitr)
library(heartrate)
library(googlesheets)
library(tidyverse)
library(forecast)
```

Here are some examples of what you can do with my packages and some basic analysis:

Sample Dataset

I created a sample dataset. It has the first 21 days of data that I collected. This time I converted it into a tibble, so it would match the rest of the data. Here is just the first couple of rows:

```
data("ckheartRate")
ckheartRate
```

```
## # A tibble: 21 x 9
##       Day `Resting Heart Rate` `Time in Min` `Miles walking` `Inf`
##     <int>                <int>         <lgl>           <dbl> <dbl>
## 1     1                   41            NA            3.30     4
## 2     2                   60            NA            1.80     4
## 3     3                   76            NA            1.90     2
## 4     4                   81            NA            3.00     1
## 5     5                   72            NA            0.55     2
## 6     6                   78            NA            0.96     3
## 7     7                   68            NA            1.80     4
## 8     8                   75            NA            1.80     0
## 9     9                   NA            NA            1.90     0
## 10   10                   69            NA            2.60     0
## # ... with 11 more rows, and 4 more variables: Computing <dbl>,
## #   Probability <dbl>, `Estimated Sleep` <dbl>, `Hrs on Hwk` <dbl>
```

Updating Dataset

I created a function that will automatically pull in the newest data. It is called updating_heartRate. This is useful so that I can continually use the function to update my data.

```
tester <- updating_heartRate()
tester
```

```
## # A tibble: 37 x 11
##          Date   Day `Resting Heart Rate` `Time in Min` `Miles walking`
##         <date> <int>                <int>         <lgl>           <dbl>
##  1 2017-09-12     1                   41            NA            3.30
##  2 2017-09-13     2                   60            NA            1.80
##  3 2017-09-14     3                   76            NA            1.90
##  4 2017-09-15     4                   81            NA            3.00
##  5 2017-09-16     5                   72            NA            0.55
##  6 2017-09-17     6                   78            NA            0.96
##  7 2017-09-18     7                   68            NA            1.80
##  8 2017-09-19     8                   75            NA            1.80
##  9 2017-09-20     9                   NA            NA            1.90
## 10 2017-09-21    10                   69            NA            2.60
## # ... with 27 more rows, and 6 more variables: `Inf` <dbl>,
## #   Computing <dbl>, Probability <dbl>, Other <int>, `Estimated
## #   Sleep` <dbl>, `Hrs on Hwk` <dbl>
```

target_ranges

I created a function that will create target work out ranges based off https://www.active.com/fitness/articles/how-to-calculate-your-training-heart-rate-zones. I think this could be useful if I tracked my heart rate during work outs I could see if my heart rates during the work out were in my target heart range. Here is an example of that:

```
ranged <- target_ranges(tester$`Resting Heart Rate`, time_data = tester$Date,
                        age = 24)
plot(ranged[,1], ranged[,2], type = "l", ylim = c(min(ranged[,2]),
                                                  max(ranged[,3])),
     xlab = "Date",
     ylab = "Heart Rate (Beats Per Min)" )
lines(ranged[,1], ranged[,3], type = "l")
```
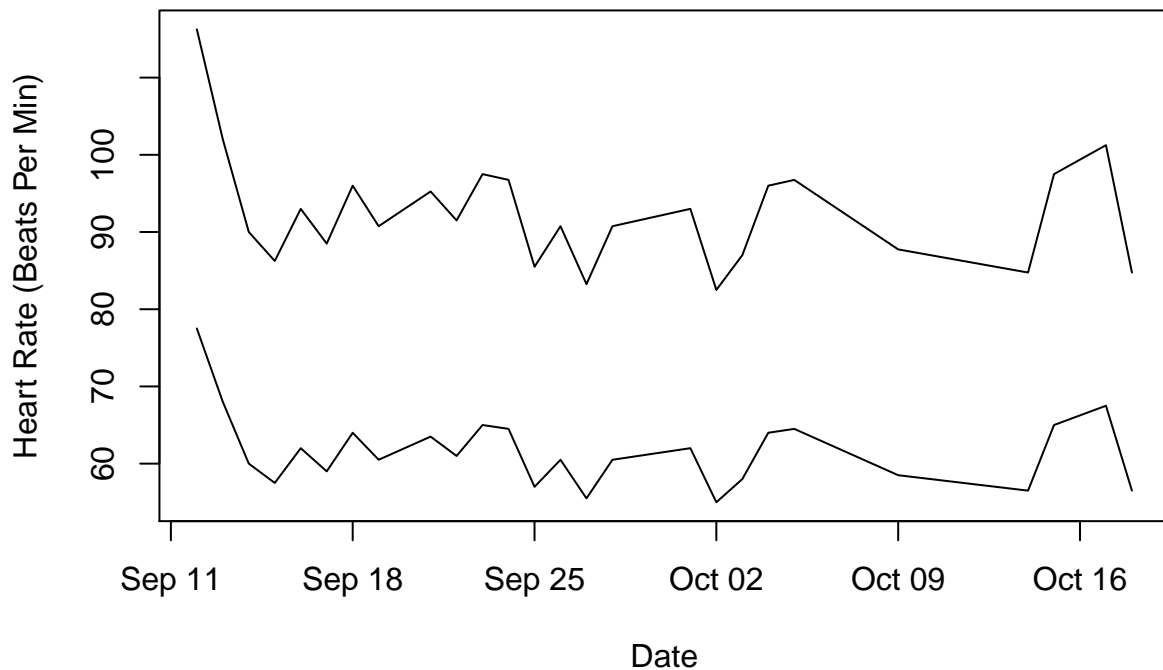
stress_plot

I created a function that imediately plots the difference between the actual data and a moving average. This would allow one to see if they seem to be above their moving average. My data for the most part seems to be pretty much the same (it might be increasing slightly) except the first data point was really low. The first data point is really low because I was using a heart rate monitor that I think is broken.
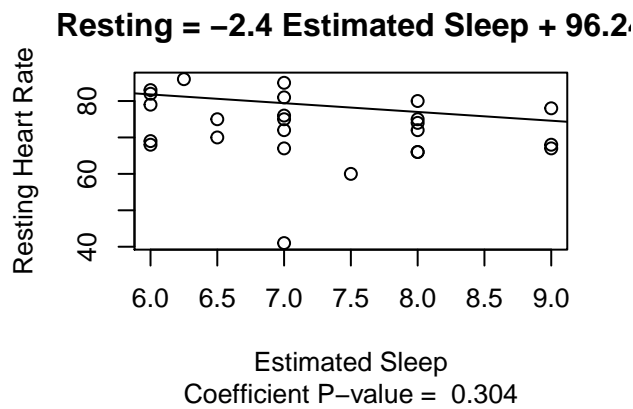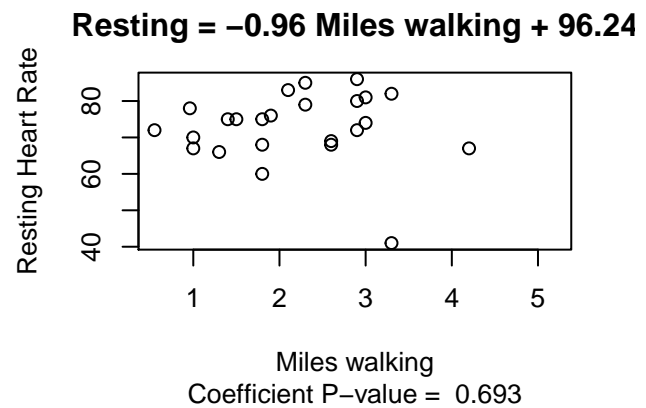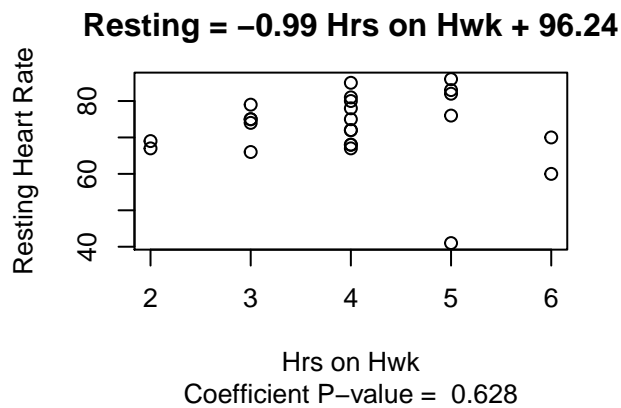
```
plot_stressed(tester$`Resting Heart Rate`, tester$Date, xlab = "Date",
              ylab = " Resting Heart Rate (Beats Per Min)")
```
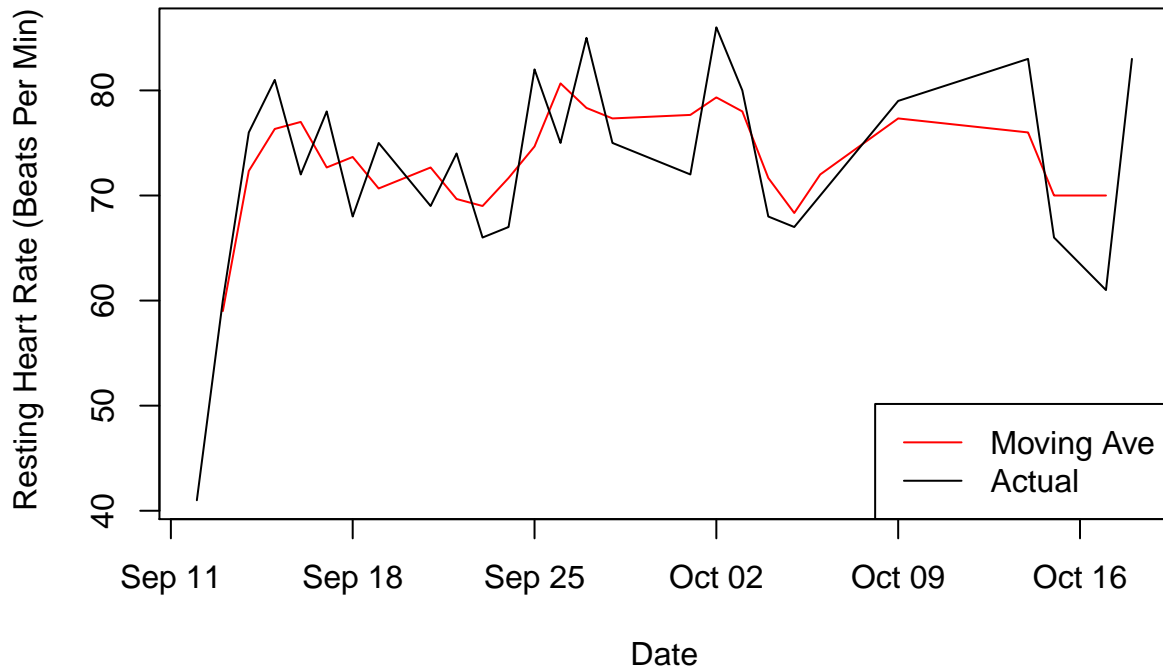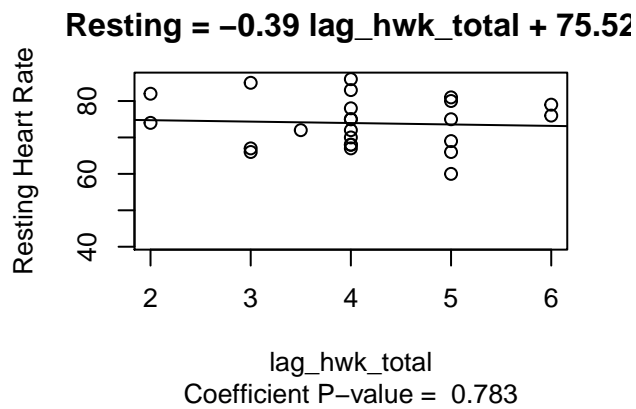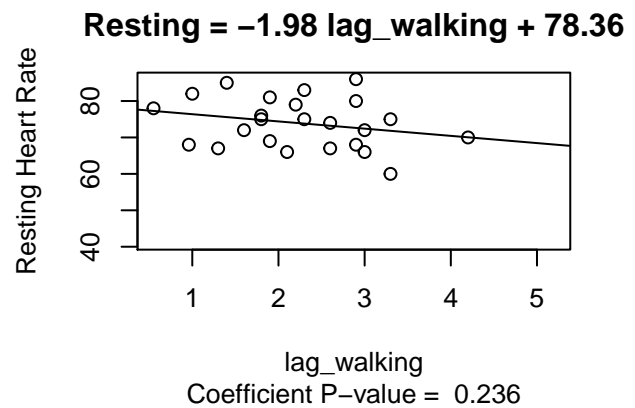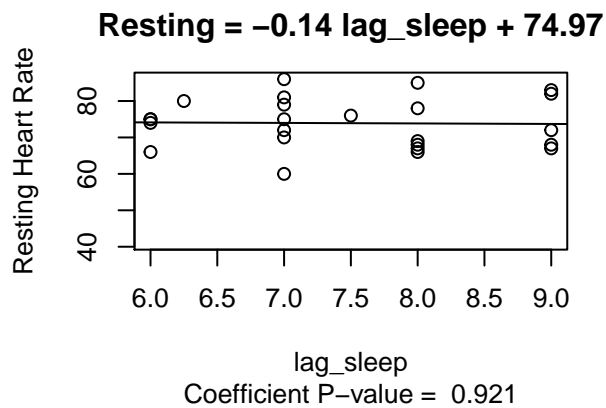
plot_lms

This function produces a series of scatterplots that match up with the columns selected. It then fits a linear regression and posts the equation for the line at the top of each scatterplot. I was also able to now add the p-value for the coefficient for the slope of the line to the plot at the bottom. I would like to add some way of doing multiple linear regression here and maybe have a way to post if the slope of the line is statistically significant. It doesn't seem that any of these linear regressions seem to fit the data very well. Also none of the p-values are significant.

```r
par(mar= c(5.0, 4.0, 3.0, 2))
tester %>% plot_lms(., col_names = c("Hrs on Hwk", "Miles walking", "Estimated Sleep"),
         response_variable = "Resting Heart Rate", multivariate = TRUE)
```

Resting = −0.99 Hrs on Hwk + 96.24

Resting = −0.96 Miles walking + 96.24

Hrs on Hwk
Coefficient P−value = 0.628

Miles walking
Coefficient P−value = 0.693

Resting = −2.4 Estimated Sleep + 96.24

Estimated Sleep
Coefficient P−value = 0.304

I then tried an compare the resting heart rate to the pervious day's data. Again, there doesn't seem to be any clear fit to the data. Also none of the p-values for the coefficient of the slope are statistically significant either.

```r
par(mar= c(5.0, 4.0, 3.0, 2))
tester <- tester %>% mutate(lag_sleep = lag(tester$"Estimated Sleep", 1),
                            lag_walking= lag(tester$"Miles walking", 1),
                            lag_hwk_total = lag(tester$"Hrs on Hwk", 1))
tester %>% plot_lms(., col_names = c("lag_sleep", "lag_walking", "lag_hwk_total"),
          response_variable = "Resting Heart Rate")
```

Resting = −0.14 lag_sleep + 74.97

lag_sleep
Coefficient P−value = 0.921

Resting = −1.98 lag_walking + 78.36

lag_walking
Coefficient P−value = 0.236

Resting = −0.39 lag_hwk_total + 75.52

lag_hwk_total
Coefficient P−value = 0.783

plot_cor

This function produces a series of scatterplots that match up with the columns selected but this time it prints the correlation of the variable of the columns selected with the response variable. As you can see for these three variables there isn't a strong correlation between them and resting heart rate.

```r
par(mar = c(4.1, 4.1,4.1,1.1))
tester %>% plot_cor(. ,col_names = c("Hrs on Hwk", "Miles walking", "Estimated Sleep"),
          response_variable = "Resting Heart Rate")
```

Correlation: –0.056

Correlation: –0.023

Correlation: –0.21
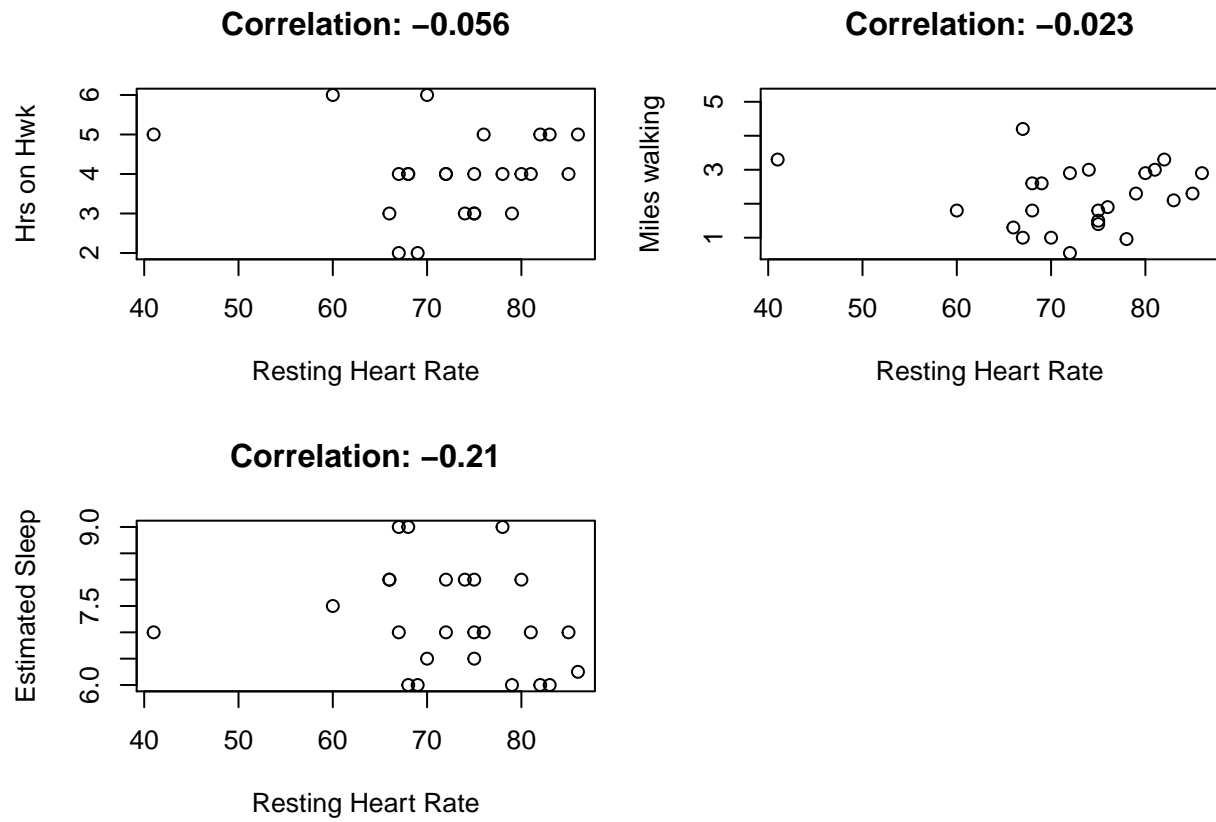
Other Analysis of the Resting Heart Rate Data.

The first point in the resting heart rate looks like it might be an outlier. Therefore, I thought about removing it. Also the first day I tried to use a heart rate monitor but I don't believe that it functions correctly.
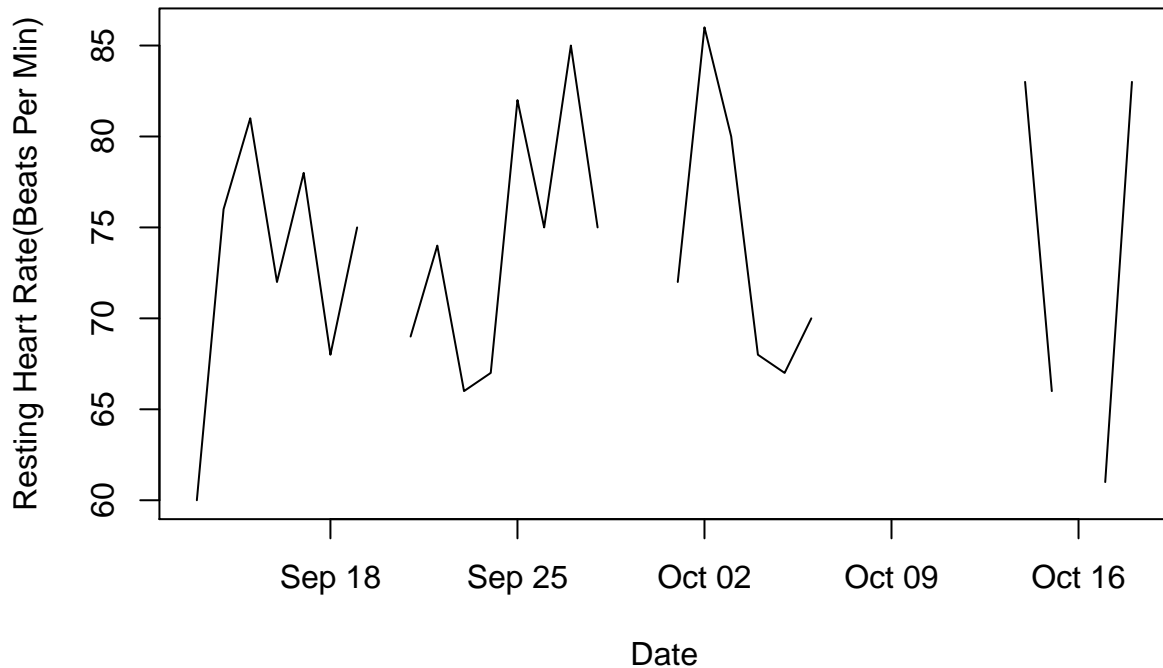
```
tester %>% select("Resting Heart Rate") %>%  boxplot()
```
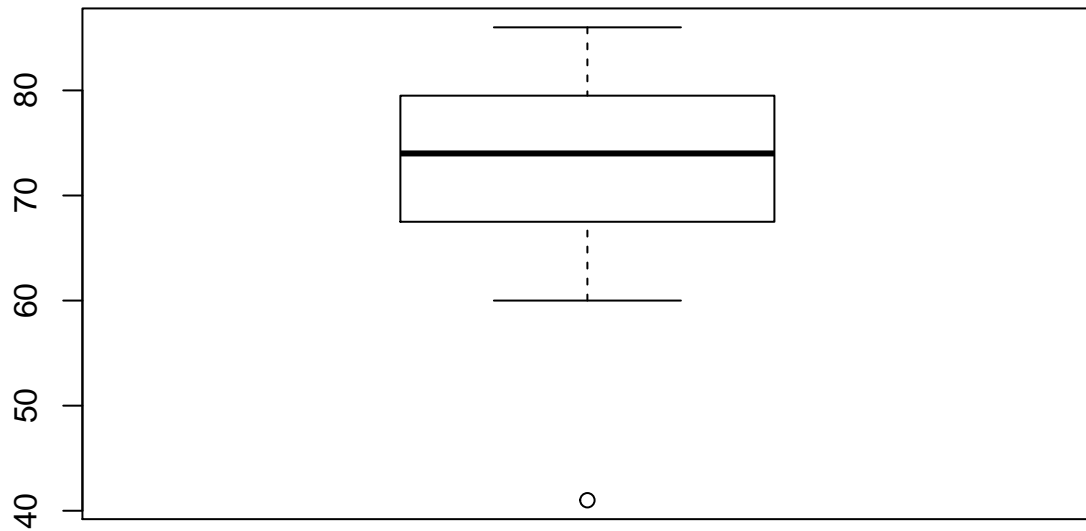
```
tester2 <- tester[-1,]
```

This is just a plot of my resting heart rate. It looks like the plot might be cyclical. So I thought to compare across days of the week.

```
plot(tester2$Date, tester2$"Resting Heart Rate", type = "l",
     ylab = "Resting Heart Rate(Beats Per Min)", xlab = "Date")
```

Interestingly, it seems that my resting heart rate is high on Monday ,Friday, Sunday. I'm not sure why that would be. This would probably something I would continually look at.

```
days_of_week <- as.factor(c("T", "W", "Th", "F", "S", "Su", "M"))
days_of_week <- rep(days_of_week, 100)[1:nrow(tester2)]
tester2$days_of_week <- days_of_week


mean_ave <- by(tester2$`Resting Heart Rate`, tester2$days_of_week, function(x) mean(x, na.rm = TRUE))
mean_ave_df <- as.data.frame(do.call(rbind, list(mean_ave)))
kable(mean_ave_df)
```
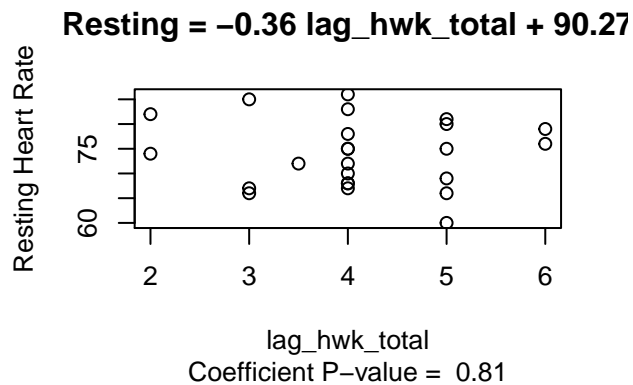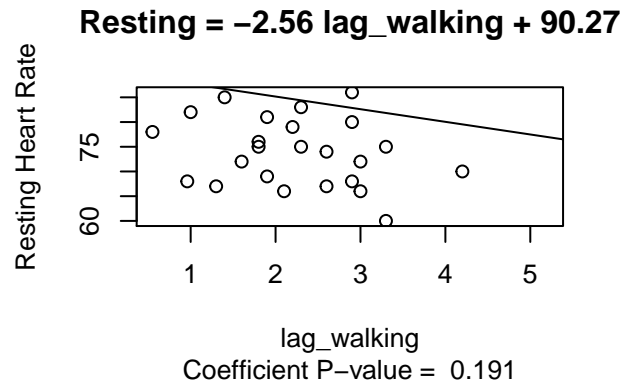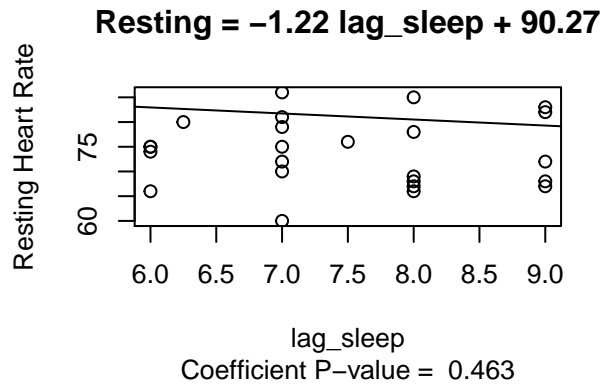
F	M	S	Su	T	Th	W
73.66667	72.75	70.75	78.75	74	75	71.75

Although, currently if we look at the the Tukey Multiple comparison test with a 95% family wise confidence level. We see that none of the of comparisons are statistically significant.

```
saver <- TukeyHSD(aov(tester2$`Resting Heart Rate` ~ tester2$days_of_week))
kable(saver$`tester$days_of_week`)
```
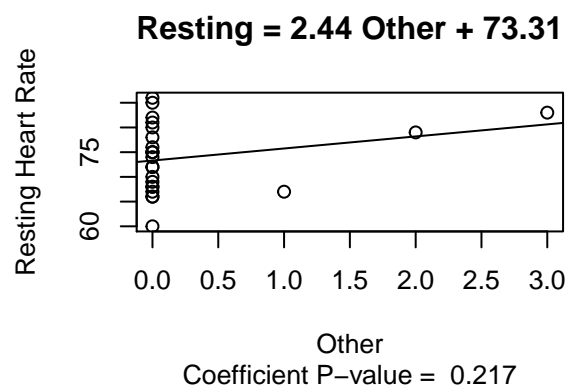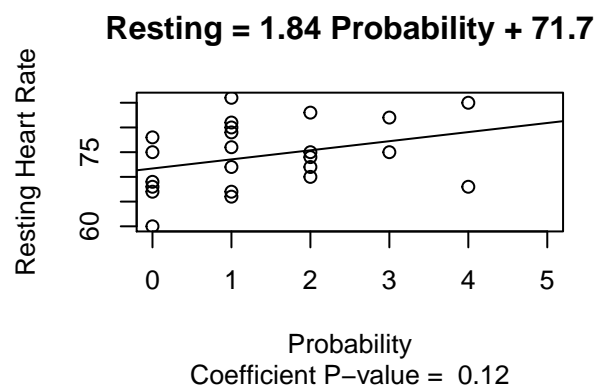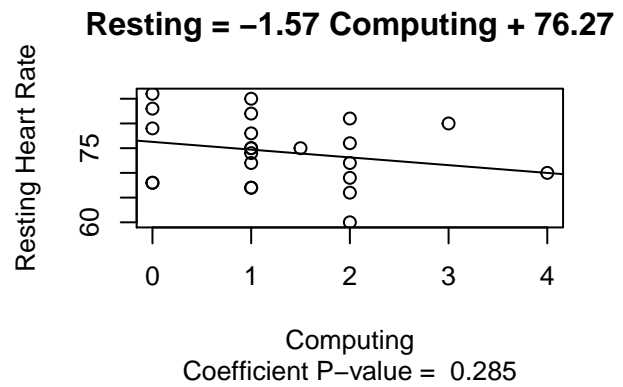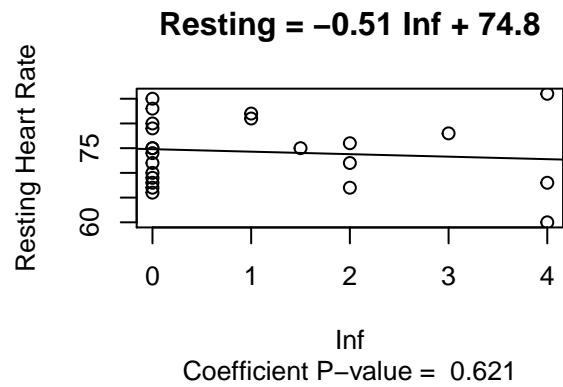
Now I though I'd look at some of the vaiables I created before but now with the dataset without the outlier and looking at the lag of sleep, miles walking, and homework totals. This time I'm going to look at them all together or in other words fit a multivariate regression model. Again, we see that none of these covariates are statistically significant.

```
tester2 %>% plot_lms(., col_names = c("lag_sleep", "lag_walking", "lag_hwk_total"),
                     response_variable = "Resting Heart Rate",
                     multivariate = TRUE)
```

Resting = −1.22 lag_sleep + 90.27

Resting Heart Rate

lag_sleep
Coefficient P−value = 0.463

Resting = −2.56 lag_walking + 90.27

Resting Heart Rate

lag_walking
Coefficient P−value = 0.191

Resting = −0.36 lag_hwk_total + 90.27

Resting Heart Rate

lag_hwk_total
Coefficient P−value = 0.81

Then I thought to just look at the indivdual classes and how much time I spend on those classes. Again I didn't see any statistically significant p-values for the correlation coefficient.
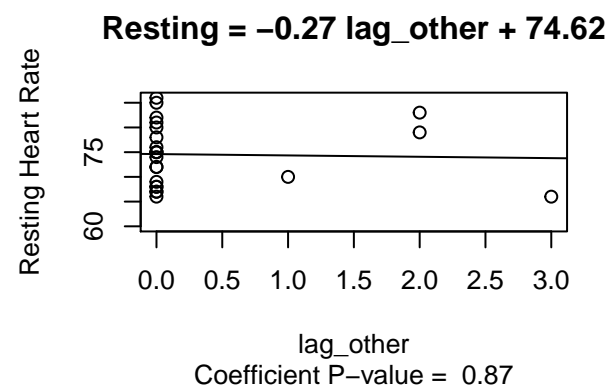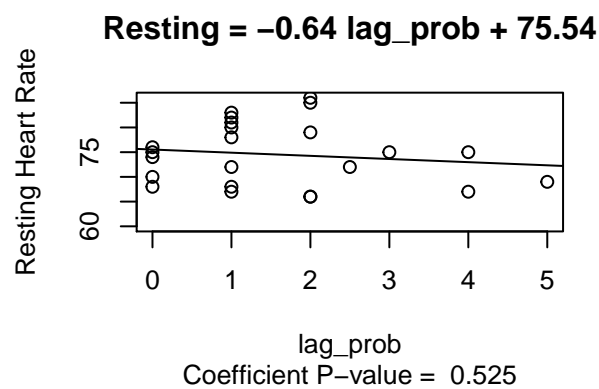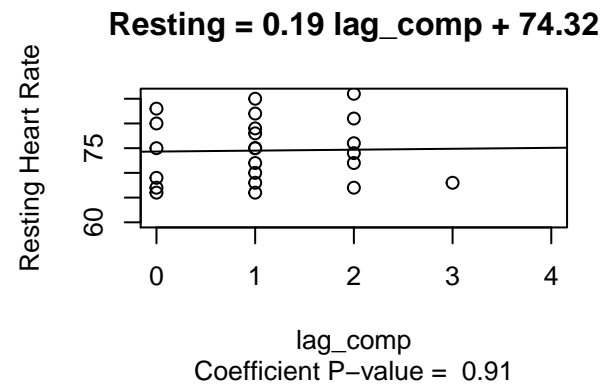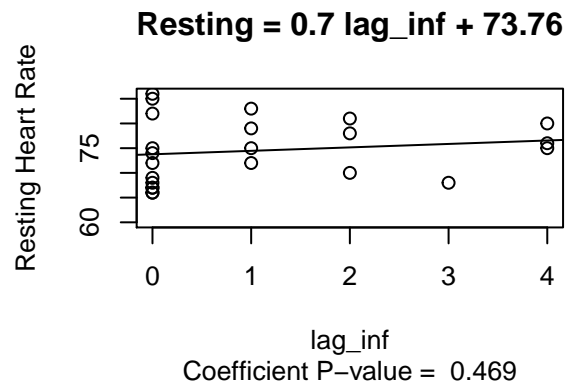
```
tester2 %>% plot_lms(., col_names = c("Inf", "Computing", "Probability", "Other"),
                     response_variable = "Resting Heart Rate")
```

Resting = –0.51 Inf + 74.8

Inf
Coefficient P–value = 0.621

Resting = –1.57 Computing + 76.27

Computing
Coefficient P–value = 0.285

Resting = 1.84 Probability + 71.7

Probability
Coefficient P–value = 0.12

Resting = 2.44 Other + 73.31

Other
Coefficient P–value = 0.217

Then again, I thought maybe if I looked at the pervious days homework for each one of those classes I would see some evidence. These variables in general seemed to even have larger p-values and not to fit the data well.

```
tester2 <- tester2 %>% mutate(lag_inf = lag(tester2$"Inf", 1),
                              lag_comp= lag(tester2$"Computing", 1),
                              lag_prob = lag(tester2$"Probability", 1),
                              lag_other = lag(tester2$Other))

tester2 %>% plot_lms(., col_names = c("lag_inf", "lag_comp", "lag_prob", "lag_other"),
                     response_variable = "Resting Heart Rate")
```

Resting = 0.7 lag_inf + 73.76

Resting Heart Rate

lag_inf
Coefficient P−value = 0.469

Resting = 0.19 lag_comp + 74.32

Resting Heart Rate

lag_comp
Coefficient P−value = 0.91

Resting = −0.64 lag_prob + 75.54

Resting Heart Rate

lag_prob
Coefficient P−value = 0.525

Resting = −0.27 lag_other + 74.62

Resting Heart Rate

lag_other
Coefficient P−value = 0.87

Answering Questions of HW 6

1) I was able to connect my r package to github. I was able to push my package there. It took a little work but it works now.
2) I did set-up git on my computer as well.
3) I synced my git with github.
4) I've been trying to improve my package. I'm a little stuck at the moment though. Do you have any suggestions on what to look at next? Right now, nothing seems to be close to significant. So I'm not sure what to do about that. I'm thinking of adding more functions or doing some time series analysis. Although, I don't know a lot about time series analysis. 5)Submitted a link.