MT5763-Project2 Report

Student ID: 180025784

Due 5 Novmber 2018

Executive Summary

The report to build a linear model that predicts Oxygen intake rates, which is a measure of aerobic fitness. The data used in this report is from the study of Rawlings (1998), *Applied Regression Analysis: A Research Tool* 2nd Edition. The final model was fitted using linear regression, diagnosed using ANOVA, AIC, qqnorm, qqline, shapiro test, ncvtest, Durbin Watson Test and updatad to a good one. The confidence intervals are provided by the bootstrapping function written by the group work of all members of Drunken Master2. The randomisation test is also carried out. The final model states that the Oxygen intake rates has a strong relationship with age, weight, runtime, restpulse and maxpulse.

Introduction

The improvement of bootstrap function completed by Drunken Master2 provided the basic code for the conduct of this project. The data used by this project is provided by Rawlings (1998).

The purpose of the investigation of this report is to seek the best model that predicts Oxygen intake rates based on a list of measurements which can be obtained easily. All the data includes the following seven variables from 31 male samples as below:

- Age: Age in years of 31 samples
- Weight: Weight in kg of 31 samples
- Oxygen: Oxygen intake rate in *ml*/kg body weight per minute
- RunTime: time for each sample to run 1.5 miles in minutes
- RestPulse: heart rate when having a rest
- RunPulse: heart rate after running
- MaxPulse: maximum heart rate during the run

The intended audience of this report should be those are statistically literate, but not familiar with computer-intensive inference. The conduct of this project are all used R 3.5.1 (R, 2018).

Methods

Due to the fact that the target audience of this report are not familiar with computer-intensive inference, the methods mainly contains the interpretation of bootstrapping and randomisation. The aim of these is to carry out effectively statistical simulations.

1 Bootstrapping function for Confidence Intervals

Bootstrapping method is a uniform sample that is put back from a given training set, in other words, whenever a sample is selected, it may be selected again and added to the training set again. The method for providing confidence intervals was present by Bradley Efron (1987). For small data set, bootstrapping works well. Boostrapping generates approximate sampling distributions of parameters to get confidence intervals

2 Randomisation Tests

In randomisation tests, *H0* refers to null hypothesis whilst *H1* represents alternative hypothesis. Randomisation tests generates parameter distributions assuming if *H0* is true and can get *p*-values. Actually, there are three main ways of randomisation *t*-test as below:

• Comparison of means of two sampled populations

H0: the means of two populations are equal

H1: the means of two populations are not equal

• Comparison of means of a sampled population that of some hypothesised value

H0: the means of the population is equal to theoretial constant

H1: the means of two populations is equal to theoretial constant

• Comparison of the variant for analysing paired observations

H0: the slope of the relationship between explanatory variable and response variable is zero - no linear association

H1: the slope of the relationship between explanatory variable and response variable is not zero - linear association exists

Results

1 Model Fitting

The initial model was fitted with all other variables, including Age, Weight, RunTime, RestPulse and MaxPulse. The coefficients of these variables regarding to Oxygen can be seen in Figure 1.

Figure 1: Coefficients of the variables in the model

Variable	Coefficient
(Intercept)	102.934
Age	-0.227
Weight	-0.074
RunTime	-2.629
RestPulse	-0.022
RunPulse	-0.37
MaxPulse	0.303

Further analysis method ANOVA was used for initial model as in Figure 2. In Figure 2, we can see the p-value of each variable regarding to Oxygen. It is clear that RunTime has the most strongly significant correlation with Oxygen while RestPulse and Weight coule be not be very related to Oxygen. Thus, these two variables would probably be abandoned when we better the model. Also, RunPulse has a comparatively strongly significant correlation and MaxPulse and Age also affect Oxygen.

Figure 2: ANOVA of the initial model

Variable	p-value
Age	0.032
Weight	0.187
RunTime	0
RestPulse	0.747
RunPulse	0.005
MaxPulse	0.036

Generally, Akaike Information Criterion (AIC) was used to select several fitted model objects to update the initial model. AIC method offered two models, one contains all six variables whose AIC score is 58.16, another one contains five variables (Age, Weight, RunTime, RunPulse and MaxPulse) whose AIC score is 56.3. Thus, AIC method automatically chose the latter one to be the new model. Then ANOVA method would be used again to check the variables of new model and see the p-value of each variable as in Figure 3. Although Weight still not seems to be sigificantly relevant with Oxygen because its p-value is more than 0.05, the other variables all shows inordinately to be related to Oxygen, which can be inferred that the new model is much better than the initial one.

Figure 3: ANOVA of the new model

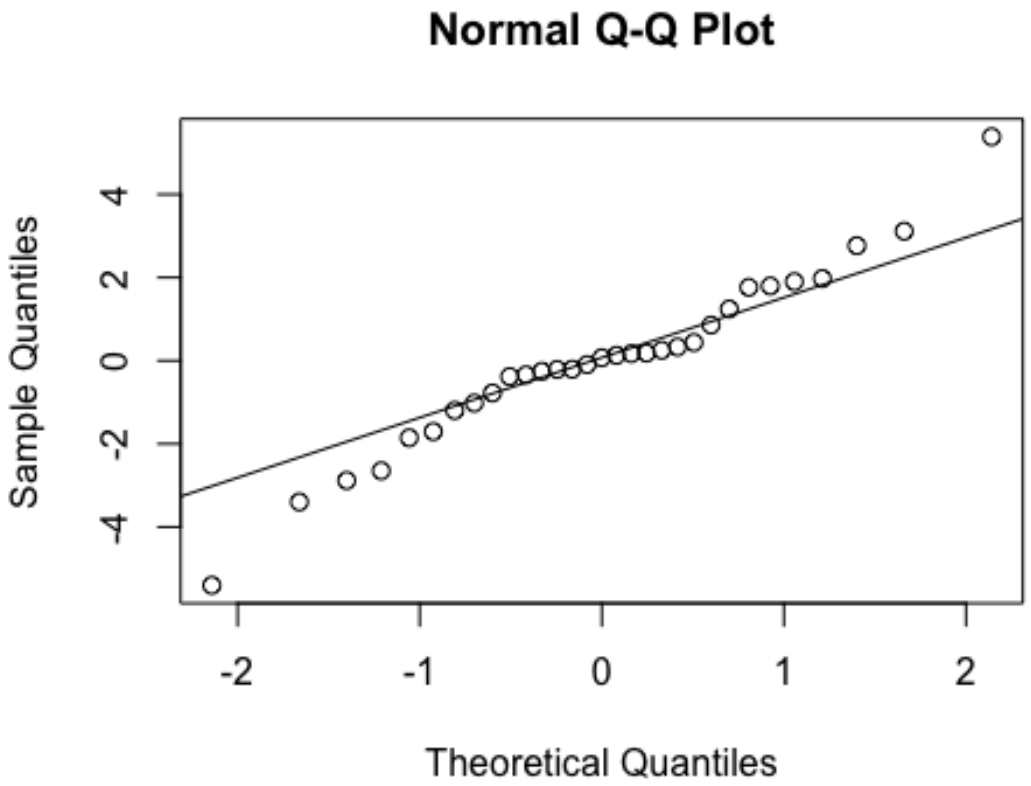
Variable	p-value
Age	0.03
Weight	0.187
RunTime	0

RunPulse 0.004

MaxPulse 0.032

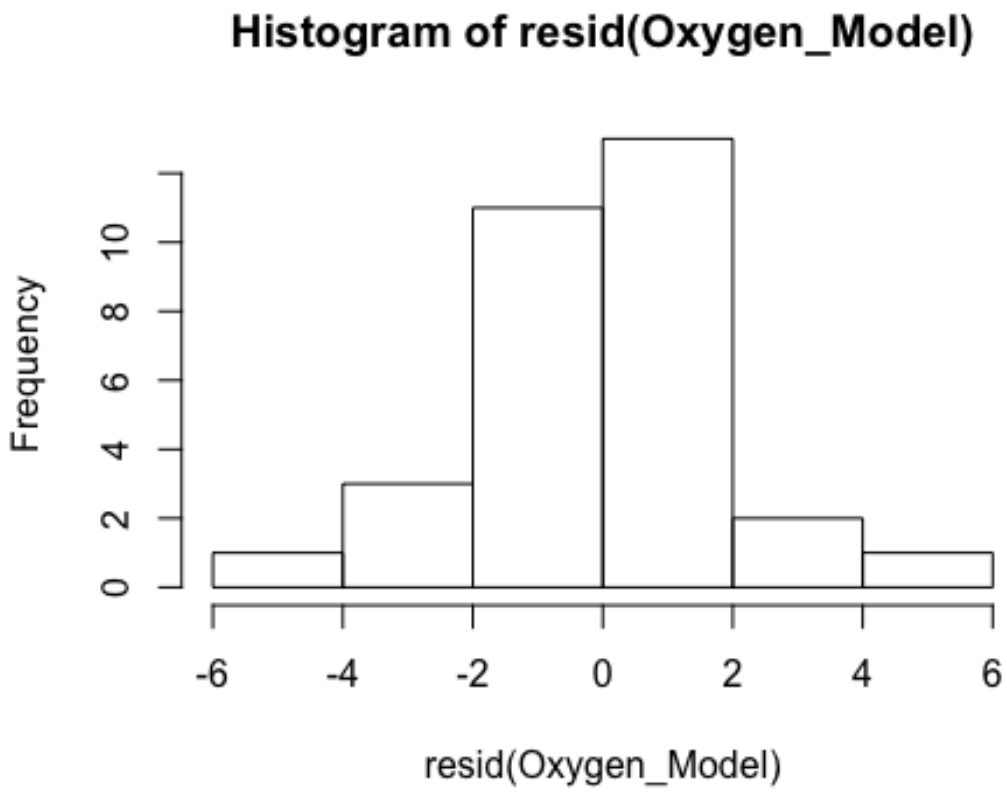
The model diagnostics was started to check the shape of errors by distribution of residuals.The QQ plot as well as the Shapiro-Wilks test was used in this report. The QQ plot and QQ norm can be seen in Figure 4 which displays the data set fits this model pretty good. Also, the Shapiro-Wilks test shows that w = 0.92131, which means the data fits well with the normal distribution in that the closer w is to 1, the model fits better with the normal distribution.

Figure 4: QQ plot of new model



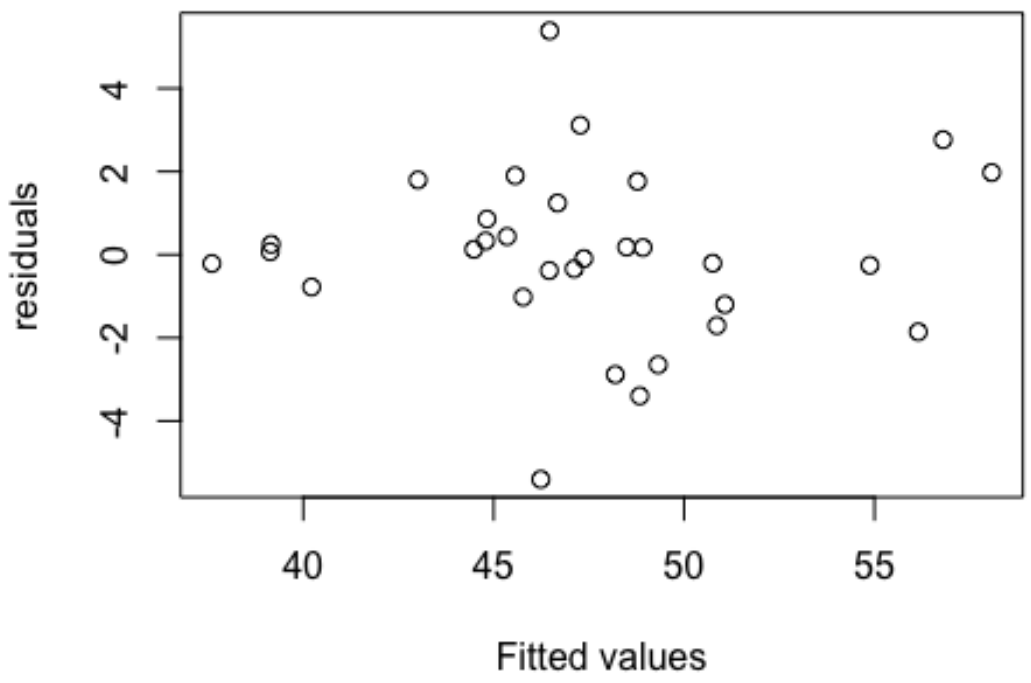
In addtion, the histogram of residuals of the new model can be seen in Figure 5. We can also track down the extreme residuals which shows that data in Row 15 and 17 are having extreme residuals.

Figure 5: histogram of residuals of new model



The next step of model diagnostics is error distribution, including check variance of residuals and Breusch-Pagan test. The error spread of the new model can be plotted in Figure 6. The output of ncvTest (i.e. Breusch-Pagan test) is p = 0.77806, which means to choose *H1* that the error variance varies with the level of the fitted value.

Figure 6: plot of residuals spread



The following step is to check serial correlation of residuals by using Durbin-Watson test. The *p*-value of this test is 0.374 > 0.05, which means the errors are correlated.

And the final step is to check collinearity of the model. Figure 7 shows the dependencies between covariates and we use this to update the model into a new altered model. In additionl, Variance Inflation Factor (VIF) was used to check if multicollinearity exists between variables in Figure 8.

Figure 7: Dependencies between covarites

```
## Loading required package: ggplot2

## ── Attaching packages ─────────────────────────────────────
tidyverse 1.2.1 ──

## ✓ tibble  1.4.2     ✓ purrr   0.2.5
## ✓ tidyr   0.8.1     ✓ dplyr   0.7.6
## ✓ readr   1.1.1     ✓ stringr 1.3.1
## ✓ tibble  1.4.2     ✓ forcats 0.3.0

## ── Conflicts ────────────────────────────────────────
tidyverse_conflicts() ──
```

```
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

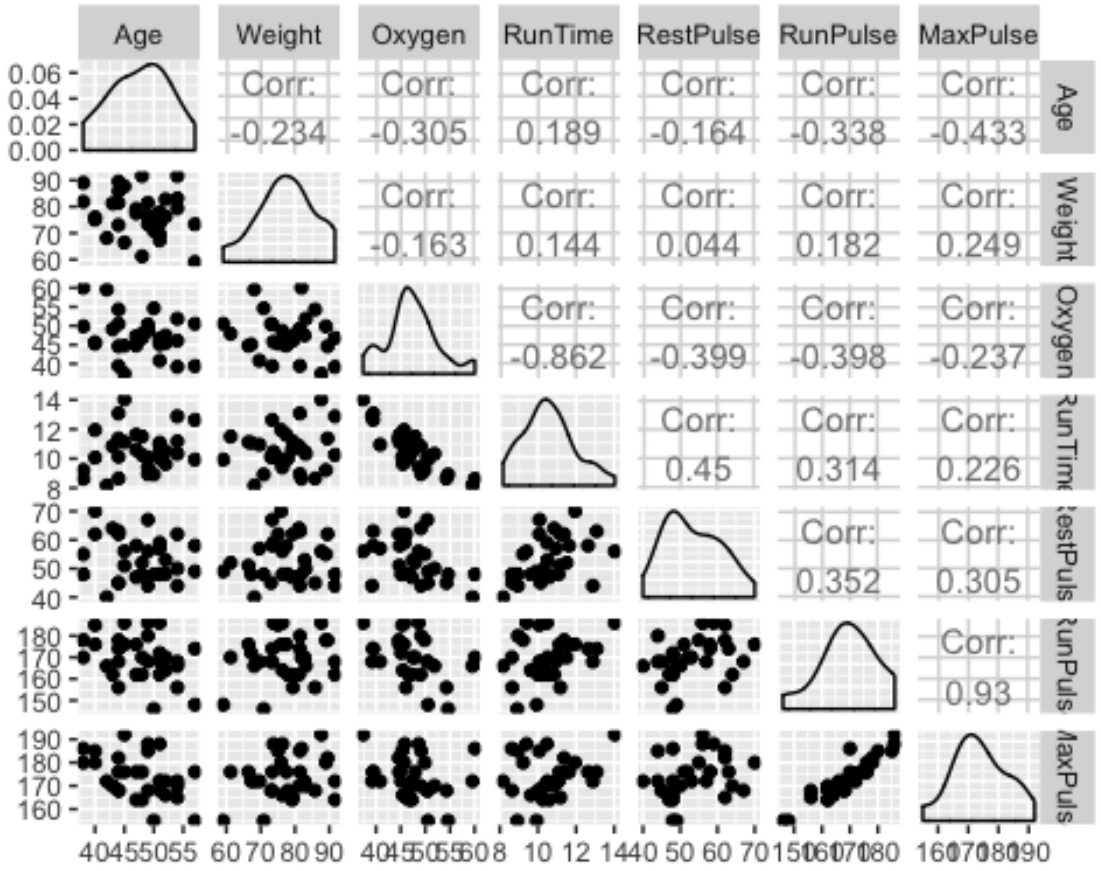
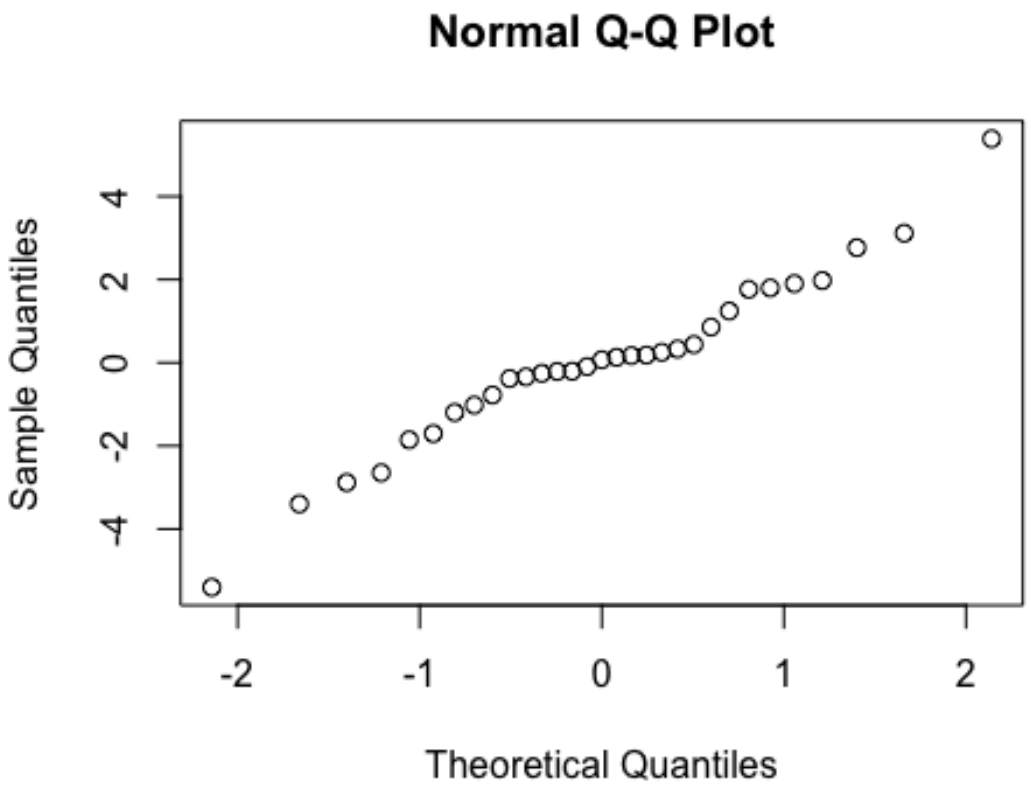


Figure 8: VIF of altered Model

Variable	VIF
Age	1.436
Weight	1.143
RunTime	1.297
RunPulse	8.359
MaxPulse	8.731

Then the final model was altered and updated through these procedures. Then QQ norm, Shapiro-Wilks Test and ncvTest would be carried out again to see if there are any changes. The QQ norm is plotted in Figure 9 below. Plus, the Shapiro-Wilks Test is 0.96627, which is closer to 1 than the previous 0.92131, which means the data fits much better with the normal distribution. And the p-value of ncvTest is 0.43469.

Figure 9: QQ norm of altered Model



The coefficients and confidence intervals of the final model can be seen in Figure 10.

Figure 10: Coefficients and Confidence Intervals of final Model

Variable	Coefficient	2.5% CI	97.5% CI
(Intercept)	102.204	77.532	126.876
Age	-0.22	-0.416	-0.223
Weight	-0.072	-0.182	0.037
RunTime	-2.683	-3.385	-1.98
RunPulse	-0.373	-0.615	-0.132
MaxPulse	0.305	0.029	0.581

2 Bootstrapping for Confidence Intervals

The bootstrapping function was from the group work of Drunken Master2. And the confidence intervals of each variable of the final model can be obtained as below in Figure 11.

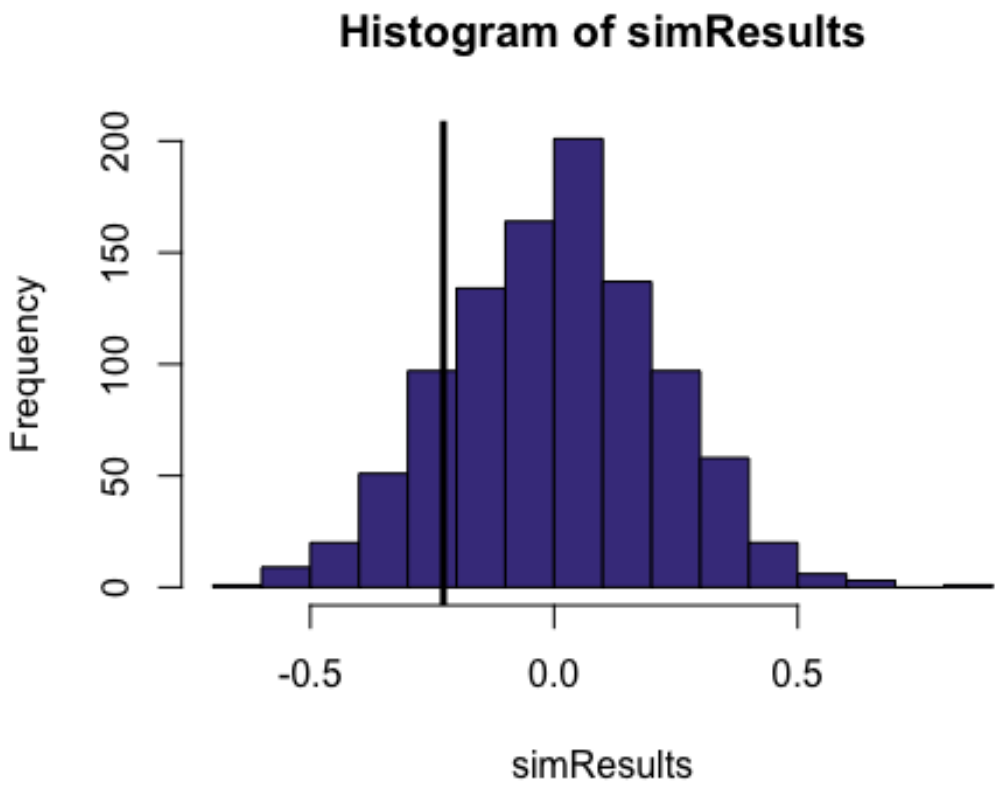
Figure 11: Confidence Intervals provided by bootstrapping function of final Model

Variable	2.5% CI	97.5% CI
(Intercept)	89	118.2
Age	-0.3698	-0.0713
Weight	-0.1631	-0.026
RunTime	-2.915	-2.071
RunPulse	-0.5703	-0.1909
MaxPulse	0.1872	0.527

3 Randomisastion Tests

In order to perform randmisation test for all model terms, I chose the test for a slope parameter to see if there is a linear association between explanatory variables and the response variable in the final model. Therefore, *H0* represents the slope of the relationship is zero, that is, there is no linear association whilst *H1* represents the slope of the relationship is not zero, that is, there is linear association. The result of this can be shown in the form of histogram with abline in Figure 12.

Figure 12: Histogram and line of estimatedSlope



From the figure, it can be observed that data is a little bit extreme if *H0* is true. Thus, maybe we should reject *H0*. To be more precise, I calculated the the minimum of $k/1000$ and $1-k/1000$, where k is the ordered position of original sample estimate. The output of this is 0.31, which means p <0.31. Therefore, it is hard to say that *H0* can be accepted. We can say from this that there is linear association in the final model.

Discussion

The model was fitted and updated to predict Oxygen intake rates on the basis of a series of other measurements. The final model is alteredModel. This model states that the parameters affecting Oxygen intake rates are Age, Weight, RunTime, RunPulse and MaxPulse.

The effects of the variables in alteredModel on Oxygen intake rates are as below:

• As the age increases by 1 year, oxygen intake rates decreases by 0.2196 ml/kg per minute

• As the weight increases by 1 kg, oxygen intake rates decreases by 0.0723 ml/kg per minute

• As the runtime increases by 1 unit, oxygen intake rates decreases by 2.6825 ml/kg per minute

• As the runpulse increases by 1 unit, oxygen intake rates decreases by 0.3734 ml/kg per minute

• As the maxpulse increases by 1 unit, oxygen intake rates increases by 0.3049 ml/kg per minute

It can be concluded from the model that elder people and heavier people will intake less oxygen rate. Also, the less the runtime and runpulse are, the less oxygen intake rates is. However, as the maxpulse increases, oxygen intake rates will increase, which is the only variable that makes oxygen intake rates high.

Additionally, the partial relationships between each covariate and the response can be found in Appendix.

References

Donovan, C. (2018) MT5763 Project 2 - code collaboration and computer intensive inference. [Online]

Efron, B. (1987) 'Better bootstrap confidence intervals', *Journal of the American Statistical Association*, 82(397), pp. 171-385. Available at:https://www.jstor.org/stable/pdf/2289144.pdf?casa_token=4iZ6QKczb9UAAAAA:I4RQeCRasPcZkONB3D1TYhkVgOVnzqEsXLW9-hjn-n8rhEPzaW5cRm2kIi28XI00vyGc3kfo_87ThMqvfvSRjdkGsYoOyMQhLFaNXm3Rsv_pKBNGRMc/ (Accessed: 3 Nov 2018)

Rawlings, J.O. (1998) *Applied regression analysis: a research tool* (2nd Edition). Berlin: Springer.

R Core Team (2018) *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. Available at: https://www.R-project.org/ (Accessed: 2 Nov 2018)

Appendices

Appendix 1 *R code and output for model fitting*

```
#input data
fitness <- read.csv("/Users/apple/Desktop/MT5763/Assignment 2/fitness.csv",
header = T)
head(fitness)

##   Age Weight Oxygen RunTime RestPulse RunPulse MaxPulse
## 1  44  89.47 44.609   11.37        62      178      182
```

```
## 2  40  75.07 45.313   10.07        62      185      185
## 3  44  85.84 54.297    8.65        45      156      168
## 4  42  68.15 59.571    8.17        40      166      172
## 5  38  89.02 49.874    9.22        55      178      180
## 6  47  77.45 44.811   11.63        58      176      176
```

```
#fit an initial model "Oxygen_Model" to predict "Oxygen"
Oxygen_Model <- lm(Oxygen~., data = fitness)
summary(Oxygen_Model)
```

```
##
## Call:
## lm(formula = Oxygen ~ ., data = fitness)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.4026 -0.8991  0.0706  1.0496  5.3847
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 102.93448   12.40326   8.299 1.64e-08 ***
## Age          -0.22697    0.09984  -2.273  0.03224 *
## Weight       -0.07418    0.05459  -1.359  0.18687
## RunTime      -2.62865    0.38456  -6.835 4.54e-07 ***
## RestPulse    -0.02153    0.06605  -0.326  0.74725
## RunPulse     -0.36963    0.11985  -3.084  0.00508 **
## MaxPulse      0.30322    0.13650   2.221  0.03601 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.317 on 24 degrees of freedom
## Multiple R-squared:  0.8487, Adjusted R-squared:  0.8108
## F-statistic: 22.43 on 6 and 24 DF,  p-value: 9.715e-09
```

```
#anova of the initial model
#install.packages("car")
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

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

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
Anova(Oxygen_Model)
```

```
## Anova Table (Type II tests)
##
## Response: Oxygen
##             Sum Sq Df F value    Pr(>F)
## Age         27.746  1  5.1685  0.032235 *
## Weight       9.911  1  1.8461  0.186866
## RunTime    250.822  1 46.7233 4.538e-07 ***
## RestPulse    0.571  1  0.1063  0.747250
## RunPulse    51.058  1  9.5111  0.005079 **
## MaxPulse    26.491  1  4.9348  0.036007 *
## Residuals 128.838 24
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#choose a model by AIC and do anova for the new model
Oxygen_Model <- step(Oxygen_Model)
```

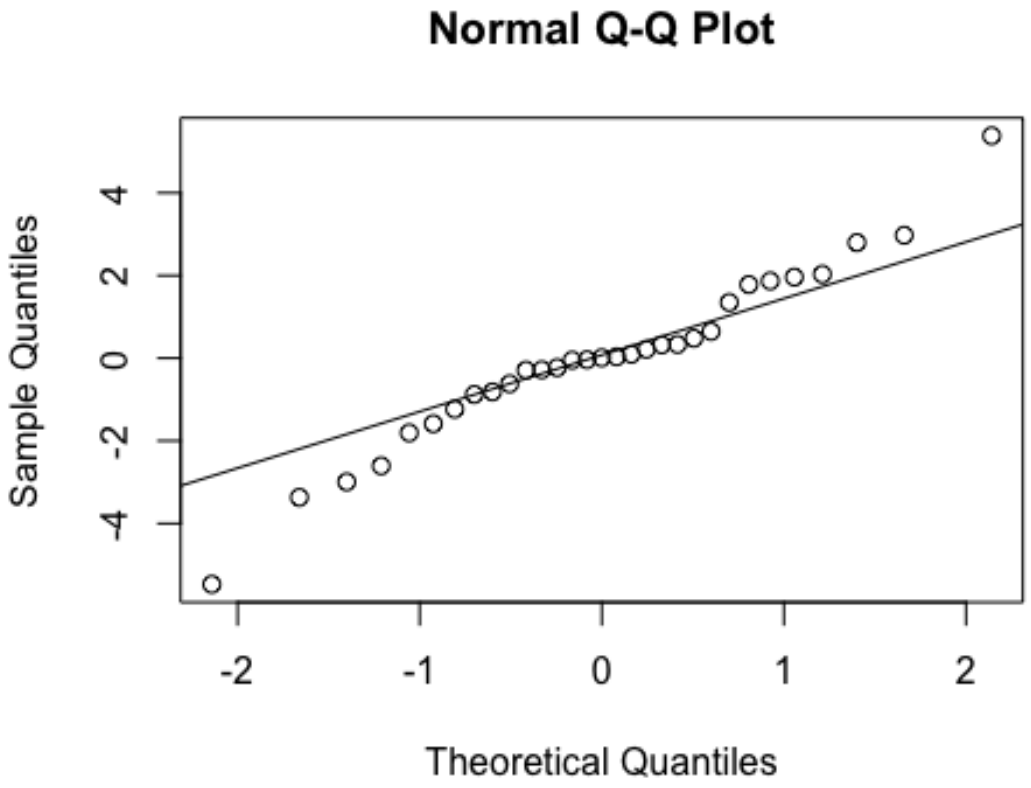
```
## Start:  AIC=58.16
## Oxygen ~ Age + Weight + RunTime + RestPulse + RunPulse + MaxPulse
##
##              Df Sum of Sq    RSS    AIC
## - RestPulse   1     0.571 129.41 56.299
## <none>                    128.84 58.162
## - Weight      1     9.911 138.75 58.459
## - MaxPulse    1    26.491 155.33 61.958
## - Age         1    27.746 156.58 62.208
## - RunPulse    1    51.058 179.90 66.510
## - RunTime     1   250.822 379.66 89.664
##
## Step:  AIC=56.3
## Oxygen ~ Age + Weight + RunTime + RunPulse + MaxPulse
##
##             Df Sum of Sq    RSS    AIC
## <none>                   129.41 56.299
## - Weight     1      9.52 138.93 56.499
## - MaxPulse   1     26.83 156.23 60.139
## - Age        1     27.37 156.78 60.247
## - RunPulse   1     52.60 182.00 64.871
## - RunTime    1    320.36 449.77 92.917
```

```
Anova(Oxygen_Model)
```

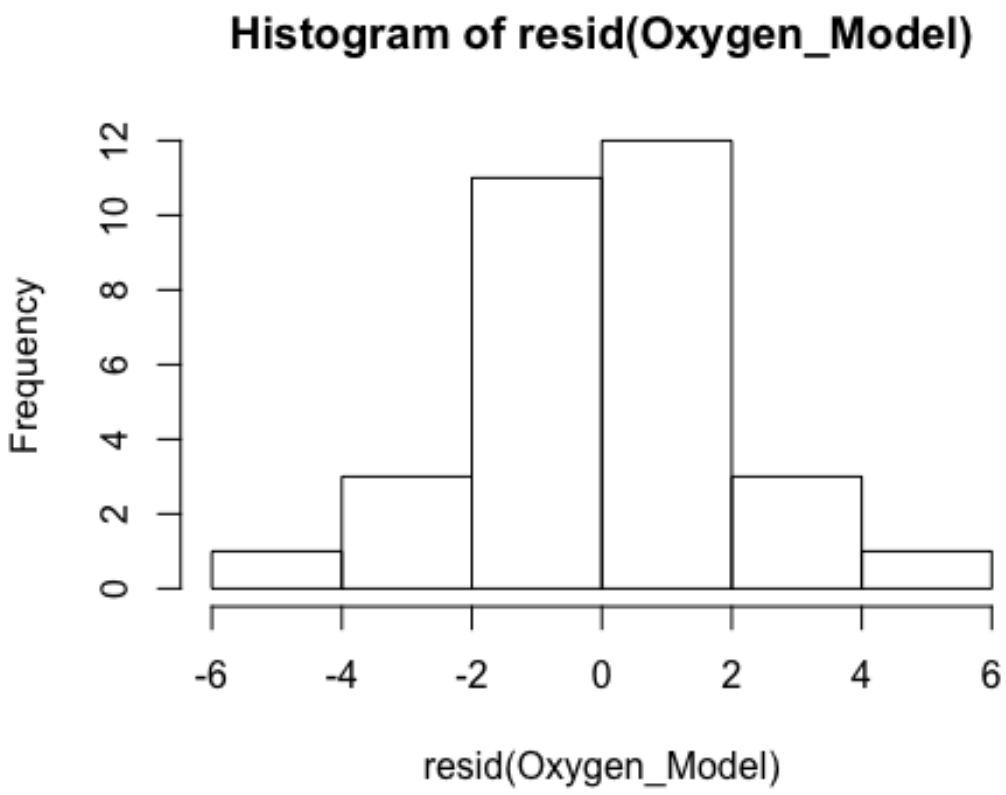
```
## Anova Table (Type II tests)
##
## Response: Oxygen
##            Sum Sq Df F value    Pr(>F)
## Age         27.37  1  5.2884   0.03010 *
## Weight       9.52  1  1.8394   0.18714
## RunTime    320.36  1 61.8892 3.186e-08 ***
## RunPulse    52.60  1 10.1609   0.00383 **
## MaxPulse    26.83  1  5.1825   0.03164 *
```

```
## Residuals 129.41 25
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

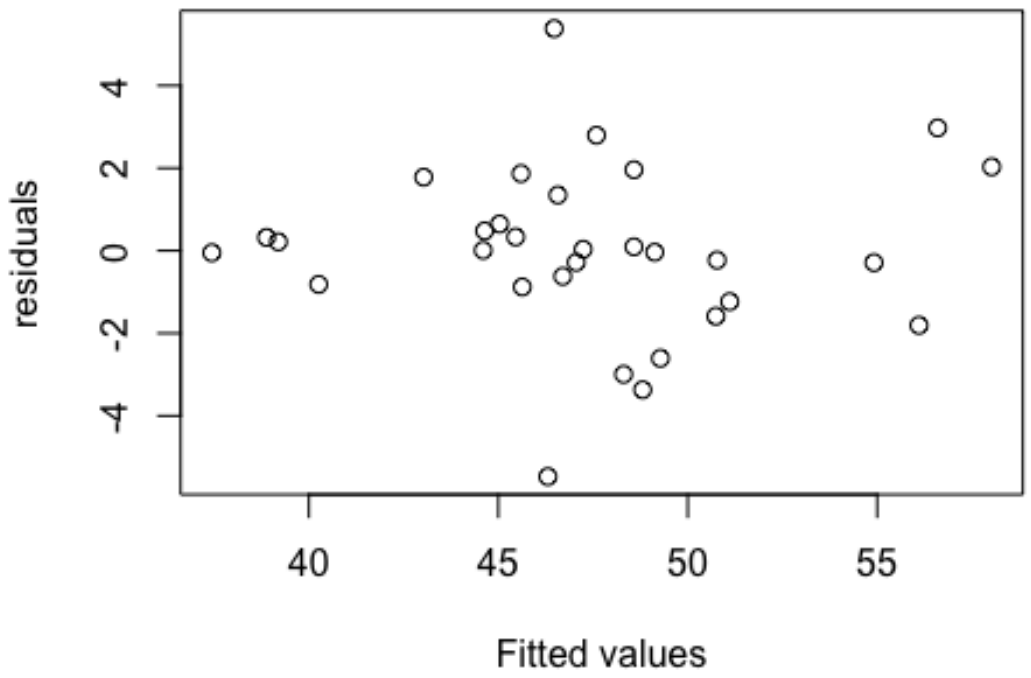
```
#model diagnostics & remedial actions

##model diagnostics - error shape
qqnorm(resid(Oxygen_Model))
qqline(resid(Oxygen_Model))
```



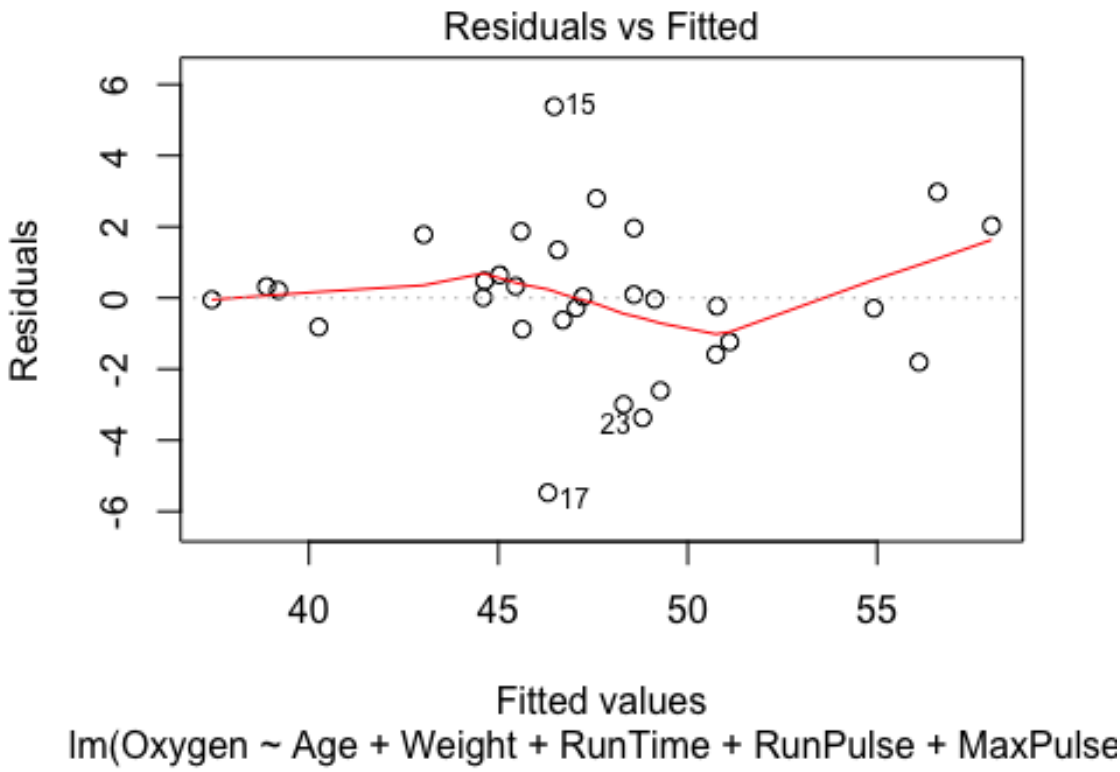

```
shapiro.test(resid(Oxygen_Model))

##
##  Shapiro-Wilk normality test
##
## data:  resid(Oxygen_Model)
## W = 0.96627, p-value = 0.4227

hist(resid(Oxygen_Model))
```

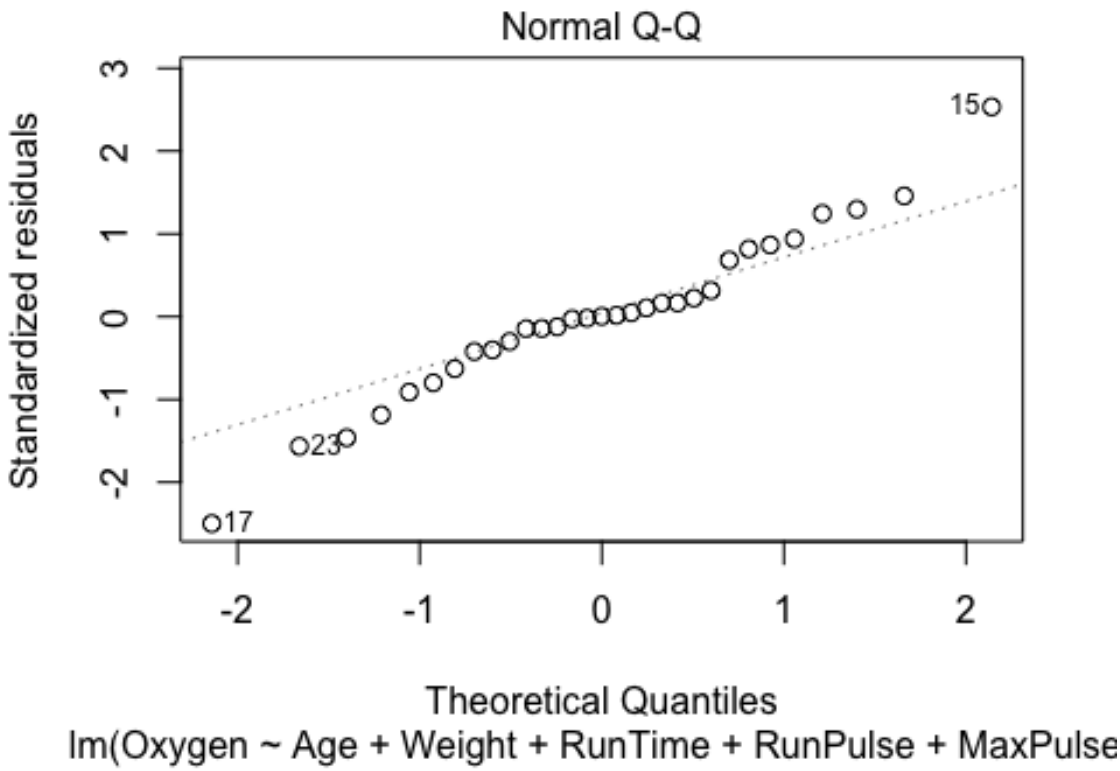



```
###track down the extreme residuals
bigResid <- which(abs(resid(Oxygen_Model))>5)
fitness[bigResid,]

##    Age Weight Oxygen RunTime RestPulse RunPulse MaxPulse
## 15  54  83.12 51.855   10.33        50      166      170
## 17  51  69.63 40.836   10.95        57      168      172

##model diagnostics - error spread
Oxygen_Resid <- resid(Oxygen_Model)
plot(fitted(Oxygen_Model), Oxygen_Resid, ylab = 'residuals', xlab = 'Fitted
values')
```

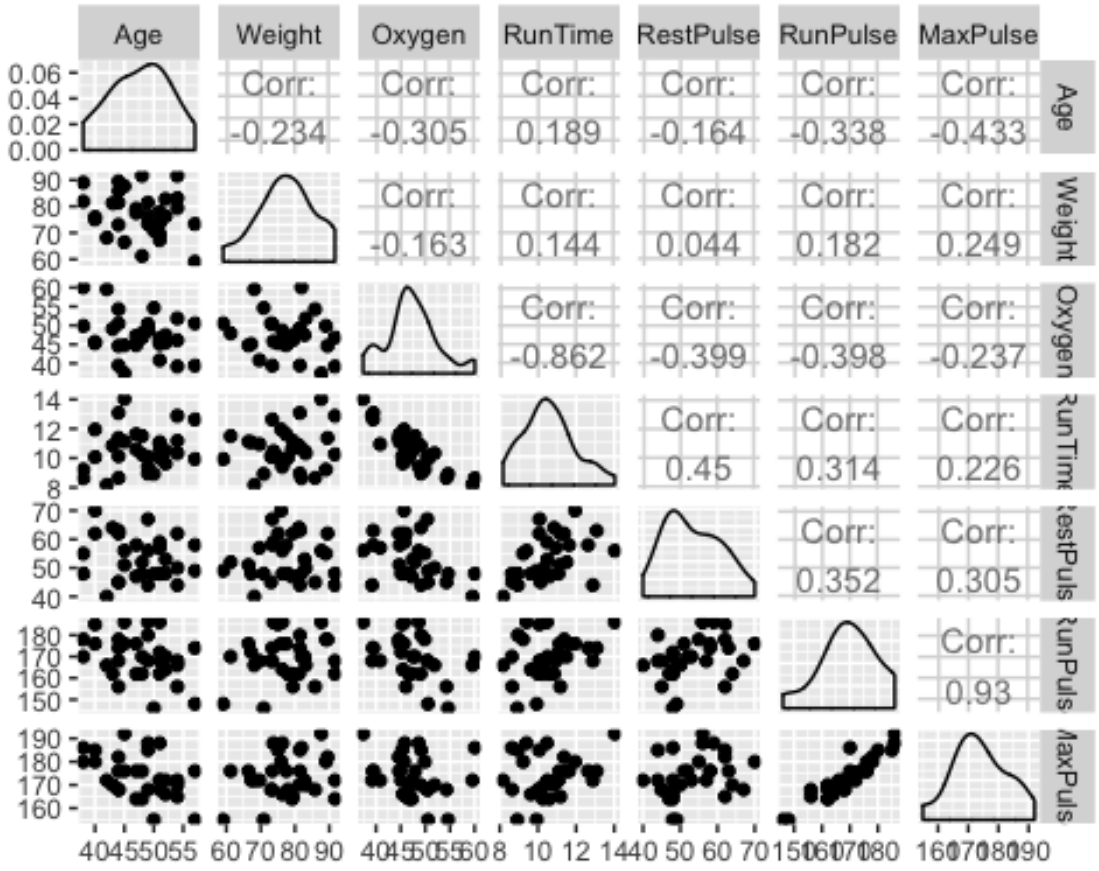



```
###testing using the ncvTest
ncvTest(Oxygen_Model, ~.)

## Non-constant Variance Score Test
## Variance formula: ~ .
## Chisquare = 4.014893, Df = 7, p = 0.77806

##model diagnostics - error independence (durbinWatsonTest)
durbinWatsonTest(Oxygen_Model)

##  lag Autocorrelation D-W Statistic p-value
##    1       0.1413284      1.690357   0.374
##  Alternative hypothesis: rho != 0

##model diagnostics - default plots
plot(Oxygen_Model, which = 1:2)
```

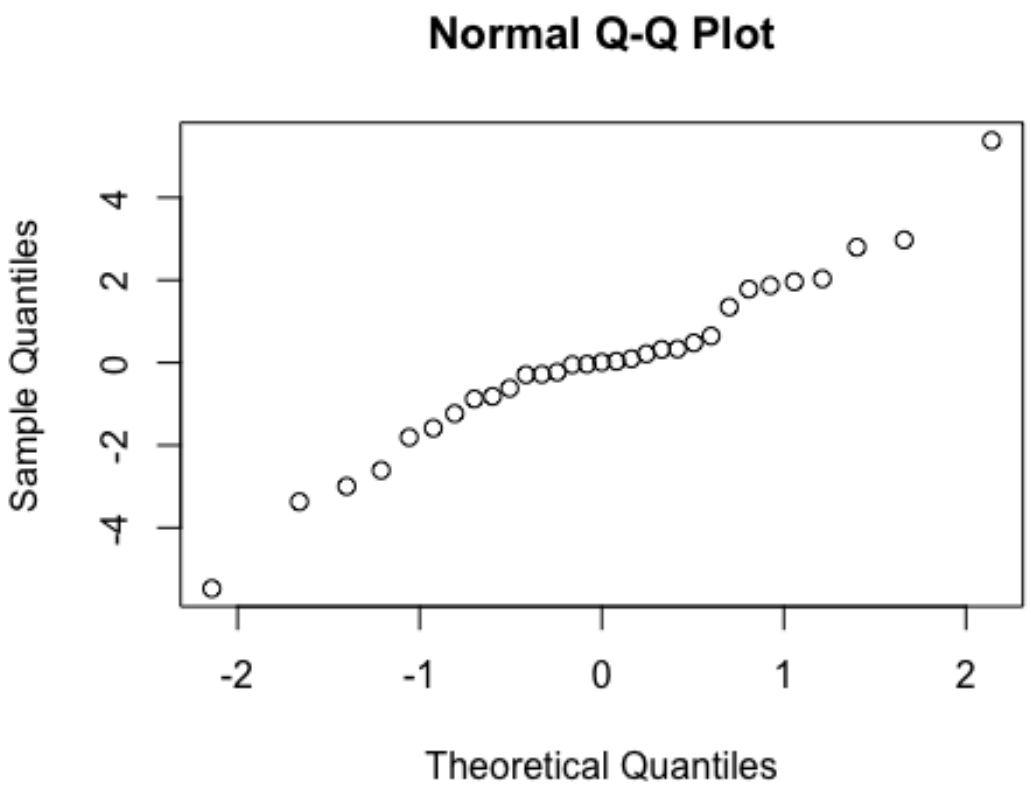

Residuals vs Fitted
Residuals
6
4
2
0
-2
-4
-6
15
23
17
40
45
50
55
Fitted values
lm(Oxygen ~ Age + Weight + RunTime + RunPulse + MaxPulse




```
##model diagnostics - collinearity
library(GGally)
library(tidyverse)
numericOnly <- fitness %>% select_if(is.numeric)
ggpairs(numericOnly)
```




```
##make a better altered model
alteredModel <- update(Oxygen_Model,.~.-comp.ratio)
vif(alteredModel)

##      Age   Weight  RunTime RunPulse MaxPulse
## 1.435634 1.142505 1.297127 8.358594 8.731225

##again: model diagnostics - error shape
qqnorm(resid(alteredModel))
```




```
shapiro.test((resid(alteredModel)))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  (resid(alteredModel))
## W = 0.96627, p-value = 0.4227
```

```
ncvTest(alteredModel)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 0.6102665, Df = 1, p = 0.43469
```

```
##summary of altered model
summary(alteredModel)
```

```
##
## Call:
## lm(formula = Oxygen ~ Age + Weight + RunTime + RunPulse + MaxPulse,
##     data = fitness)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -5.4724 -0.8476  0.0094  0.9976  5.3807
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 102.20428   11.97929   8.532 7.13e-09 ***
## Age          -0.21962    0.09550  -2.300  0.03010 *
## Weight       -0.07230    0.05331  -1.356  0.18714
## RunTime      -2.68252    0.34099  -7.867 3.19e-08 ***
## RunPulse     -0.37340    0.11714  -3.188  0.00383 **
## MaxPulse      0.30491    0.13394   2.277  0.03164 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.275 on 25 degrees of freedom
## Multiple R-squared:  0.848,  Adjusted R-squared:  0.8176
## F-statistic:  27.9 on 5 and 25 DF,  p-value: 1.811e-09

##anova and confidence intervals of altered model
Anova(alteredModel)

## Anova Table (Type II tests)
##
## Response: Oxygen
##           Sum Sq Df F value    Pr(>F)
## Age        27.37  1  5.2884   0.03010 *
## Weight      9.52  1  1.8394   0.18714
## RunTime   320.36  1 61.8892 3.186e-08 ***
## RunPulse   52.60  1 10.1609   0.00383 **
## MaxPulse   26.83  1  5.1825   0.03164 *
## Residuals 129.41 25
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

confint(alteredModel)

##                    2.5 %       97.5 %
## (Intercept) 77.53246619 126.87608420
## Age         -0.41631235  -0.02293041
## Weight      -0.18209653   0.03749185
## RunTime     -3.38479564  -1.98025030
## RunPulse    -0.61465745  -0.13214425
## MaxPulse     0.02906062   0.58075505
```

Appendix 2 *R code and output for bootstrapping code to provide confidence intervals*

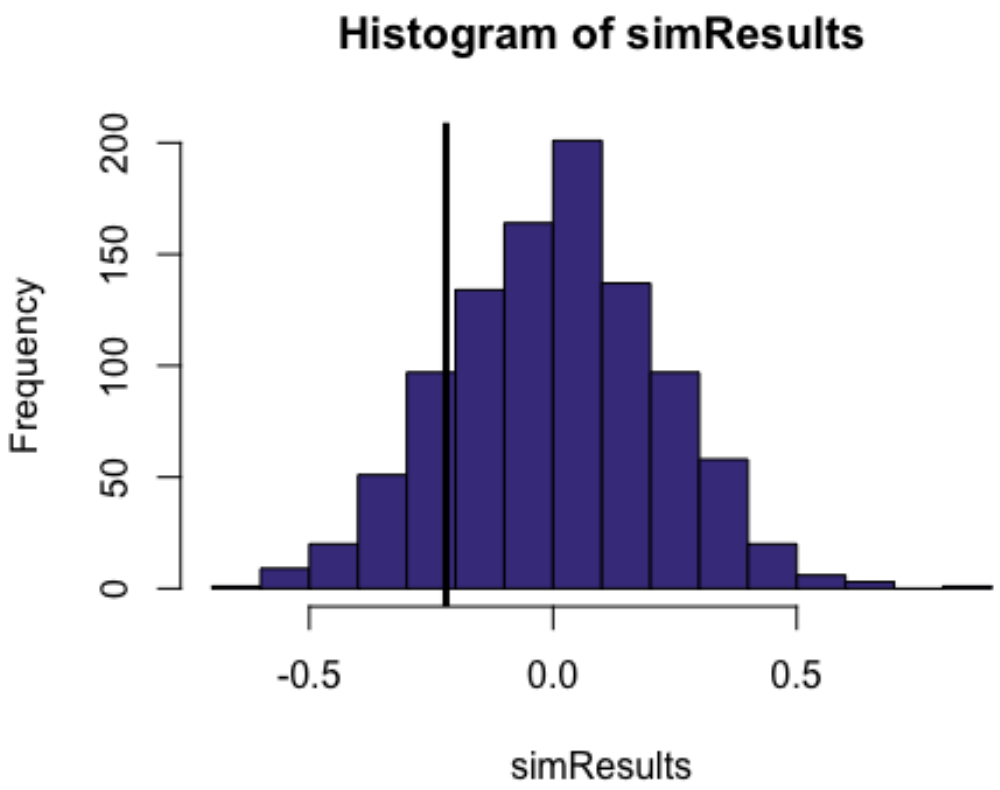
```
#the bootstrapping function from groupwork
#install.packages("boot")
library(boot)
```

```r
lmBoot_par <- function(inputData, nBoot){
  #Purpose: Generate a large number of linear regression beta coefficients
using
  #         bootstrap methods.
  #Inputs: inputData: a dataframe containing the response variable, which
must be
  #        in the first column of the dataframe, and the covariates of
interest
  #        nBoot: the number of bootstrap samples to generate.
  #Outputs: BootResults: An arraycontaing the parameter estimates of each
  #         each bootstrap sample.
  #         ConfidenceIntervals: A matrix containing 95% confidence intervals
  #         for each parameter.

  #Calculate the number of observations in the dataset
  nObs <- nrow(inputData)

  #Create the sample data with 1s for the intercept
  sampleData <- as.matrix(cbind(inputData[, 1], 1, inputData[, -1]))

  #Set up parallisation
  nCores <- detectCores()
  myClust <- makeCluster(nCores - 1, type = "PSOCK")
  registerDoParallel(myClust)

  # Create the samples
  bootSamples <- matrix(sample(1:nrow(inputData), nObs * nBoot, replace = T),
                        nrow = nObs, ncol = nBoot)

  #Use parallised sapply to apply bootLM to bootResults matrix
  bootResults <- matrix(NA, nBoot, ncol(sampleData[, -1]))
  bootResults <- parSapply(myClust, 1:nBoot, bootLM, inputData = sampleData,
                           samples = bootSamples)

  #Close parallisation
  stopCluster(myClust)

  return(t(bootResults))
}

#results of the bootstrapping results
#install.packages("parallel")
#install.packages("doParallel")
#install.packages("IPSUR")
library(parallel)
library(doParallel)
library(IPSUR)
results <- boot(data = fitness, statistic = lmBoot_par, R = 10, formula =
```

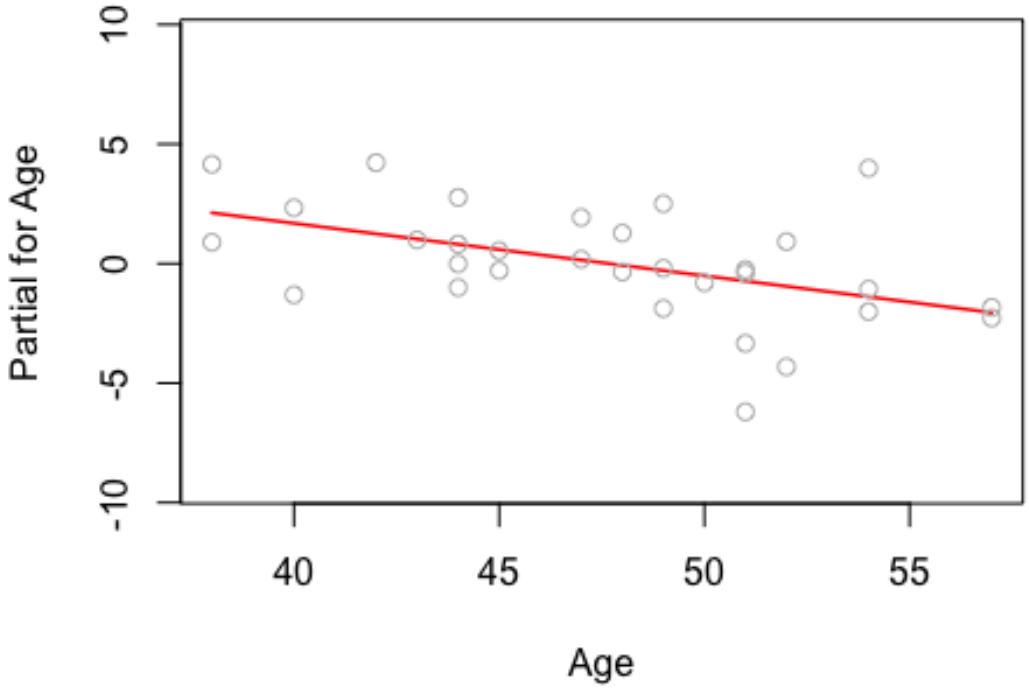
```
alteredModel)

#provide confidence intervals
boot.ci(results, type = "basic", index=1) #intercept
boot.ci(results, type = "basic", index=2) #Age
boot.ci(results, type = "basic", index=3) #Weight
boot.ci(results, type = "basic", index=4) #RunTime
boot.ci(results, type = "basic", index=5) #RunPulse
boot.ci(results, type = "basic", index=6) #MaxPulse
```

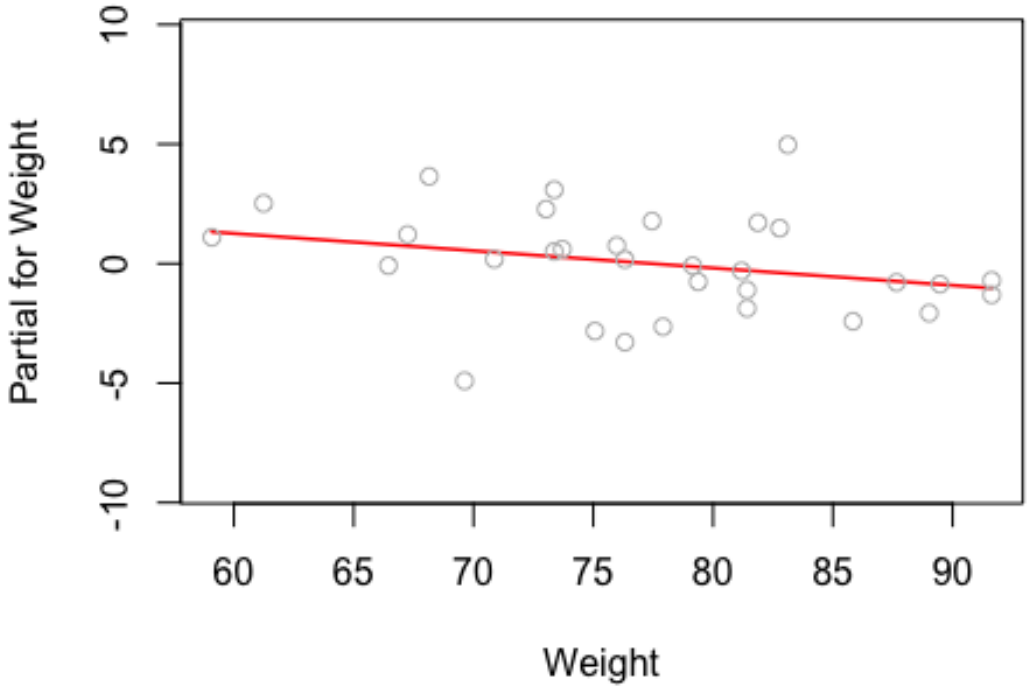
Appendix 3 *R code and output for randomisation tests*

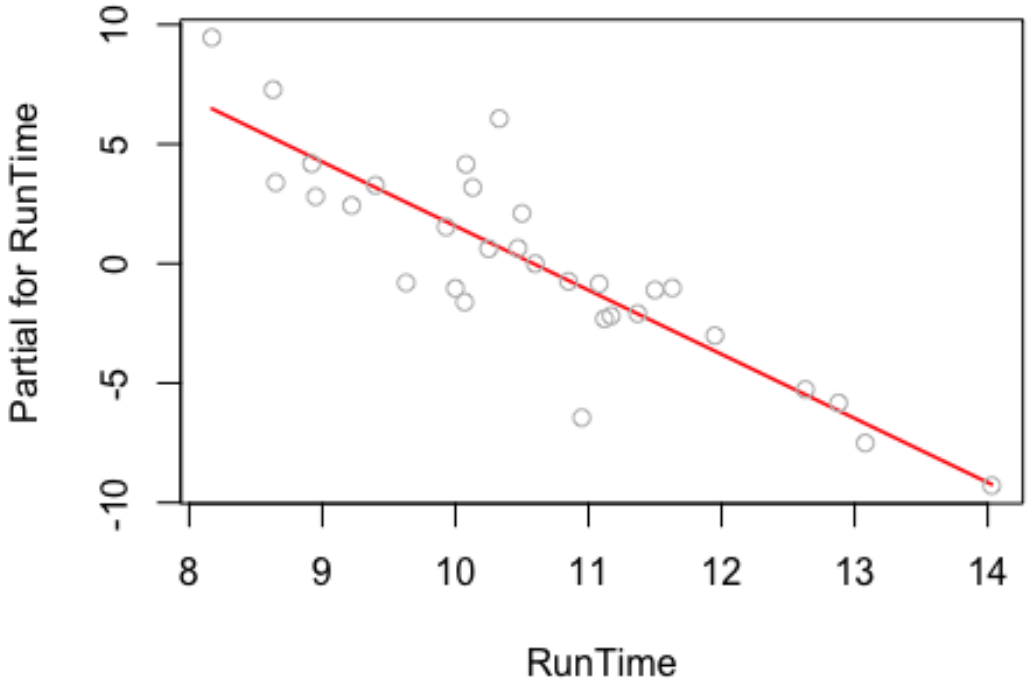
```
set.seed(180025784)

#remove the parameter "RestPulse" that is useless for the model
alteredFitness <- fitness %>% select (-RestPulse)

#compute the slope of alteredModel
estimatedSlope <- coef(alteredModel)[2]

#to store simulations
simResults <- numeric(999)

simData <- alteredFitness

for (i in 1: 999){
  #shuffle the variables WRT Oxygen
  simData$Oxygen <- sample(alteredFitness$Oxygen, 31, replace = T)
  #fit a model under H0
  simLM <- lm (Oxygen~., data = simData)
  #store the slope
  simResults[i] <- coef(simLM)[2]
}

#draw the histogram and line of estimatedSlope to compare
hist(simResults, col = "slateblue4")
abline(v = estimatedSlope, lwd = 3)
```

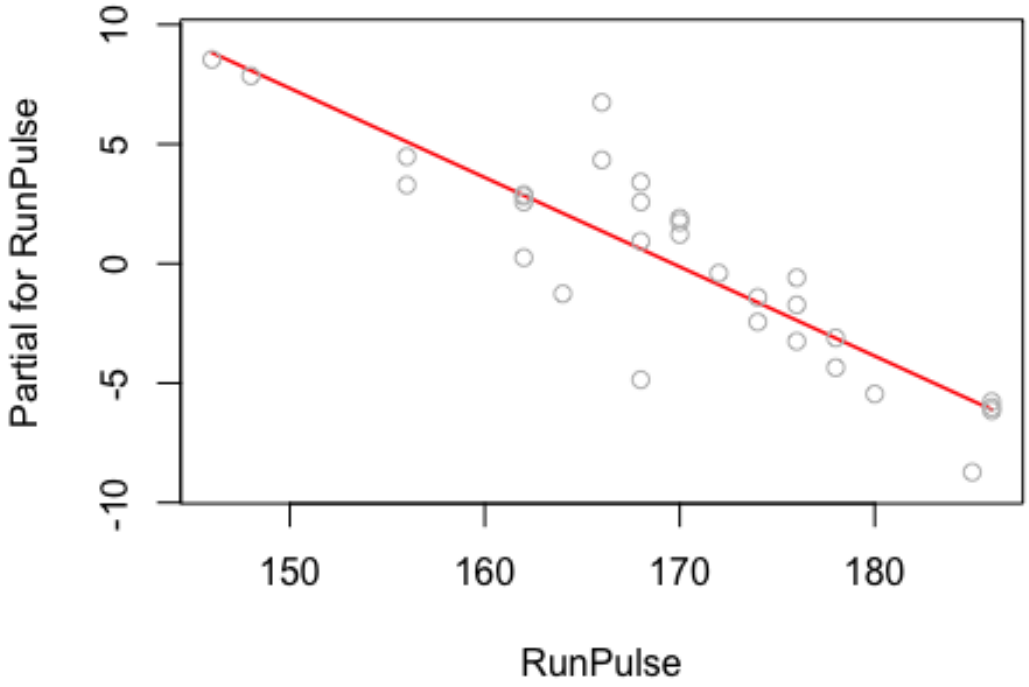


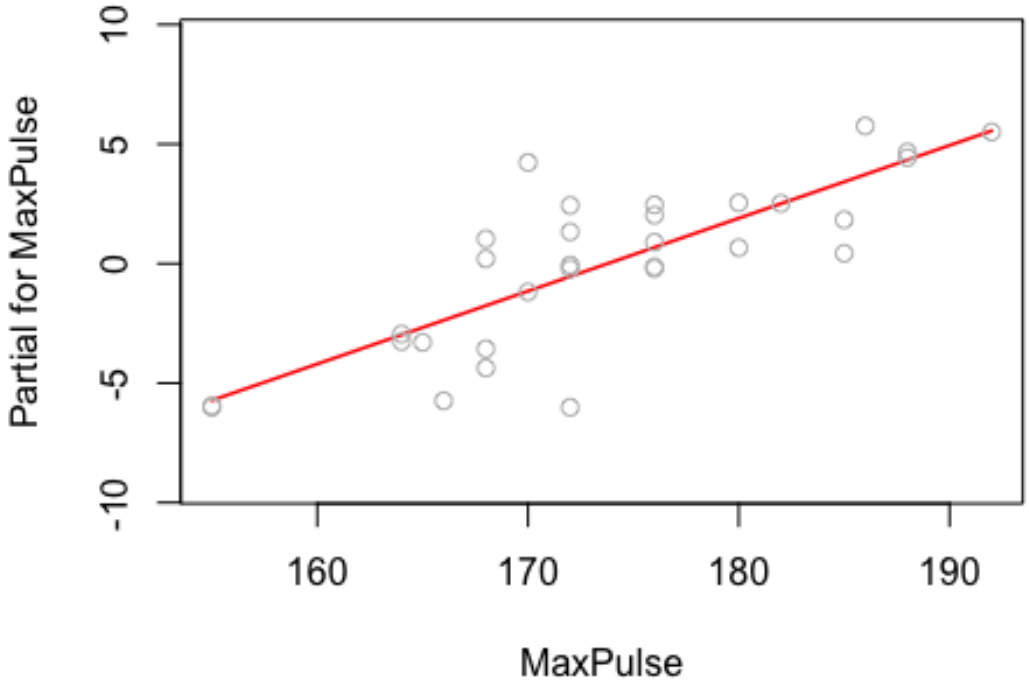

```
#reject H0 maybe

addEst <- c(estimatedSlope, simResults)

locEst <- c(1, rep(0,999))

locEst <- locEst[order(addEst)]

k <- which(locEst == 1)

min(k/1000, 1-k/1000)*2

## [1] 0.31

###partial relationships
termplot(alteredModel, data = alteredFitness,partial.resid = TRUE)
```


Partial for Age
10
5
0
-5
-10
40
45
50
55
Age


Partial for Weight
10
5
0
-5
-10
60
65
70
75
80
85
90
Weight


Partial for RunTime
10
5
0
-5
-10
8
9
10
11
12
13
14
RunTime


Partial for RunPulse
10
5
0
-5
-10
150
160
170
180
RunPulse


Partial for MaxPulse
10
5
0
-5
-10
160
170
180
190
MaxPulse