How Training Load Predicts Body Fatigue in Collegiate Soccer

Virginia Tech

Sam Rizzuto

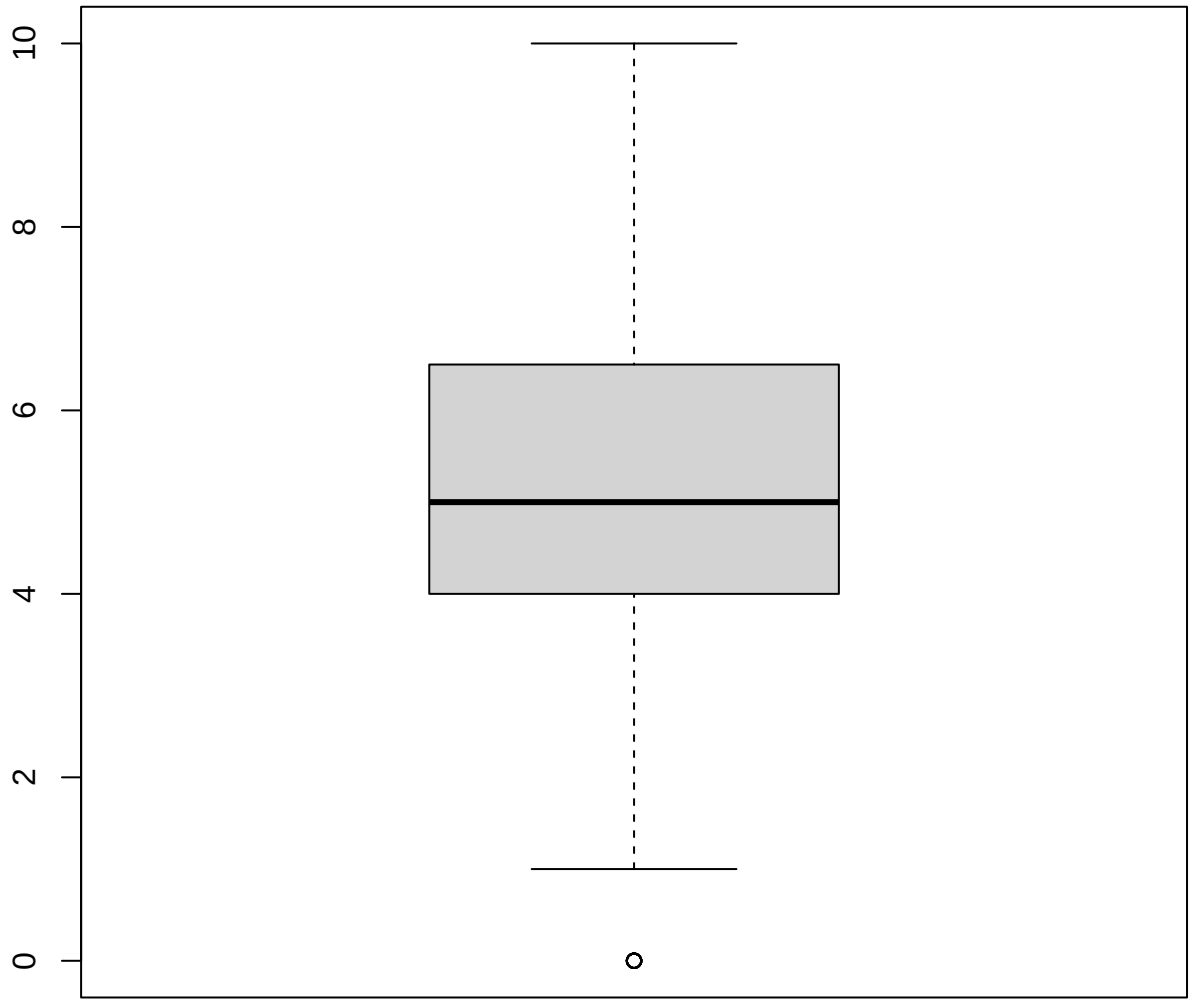
15 December 2020

Loading Necessary Libraries

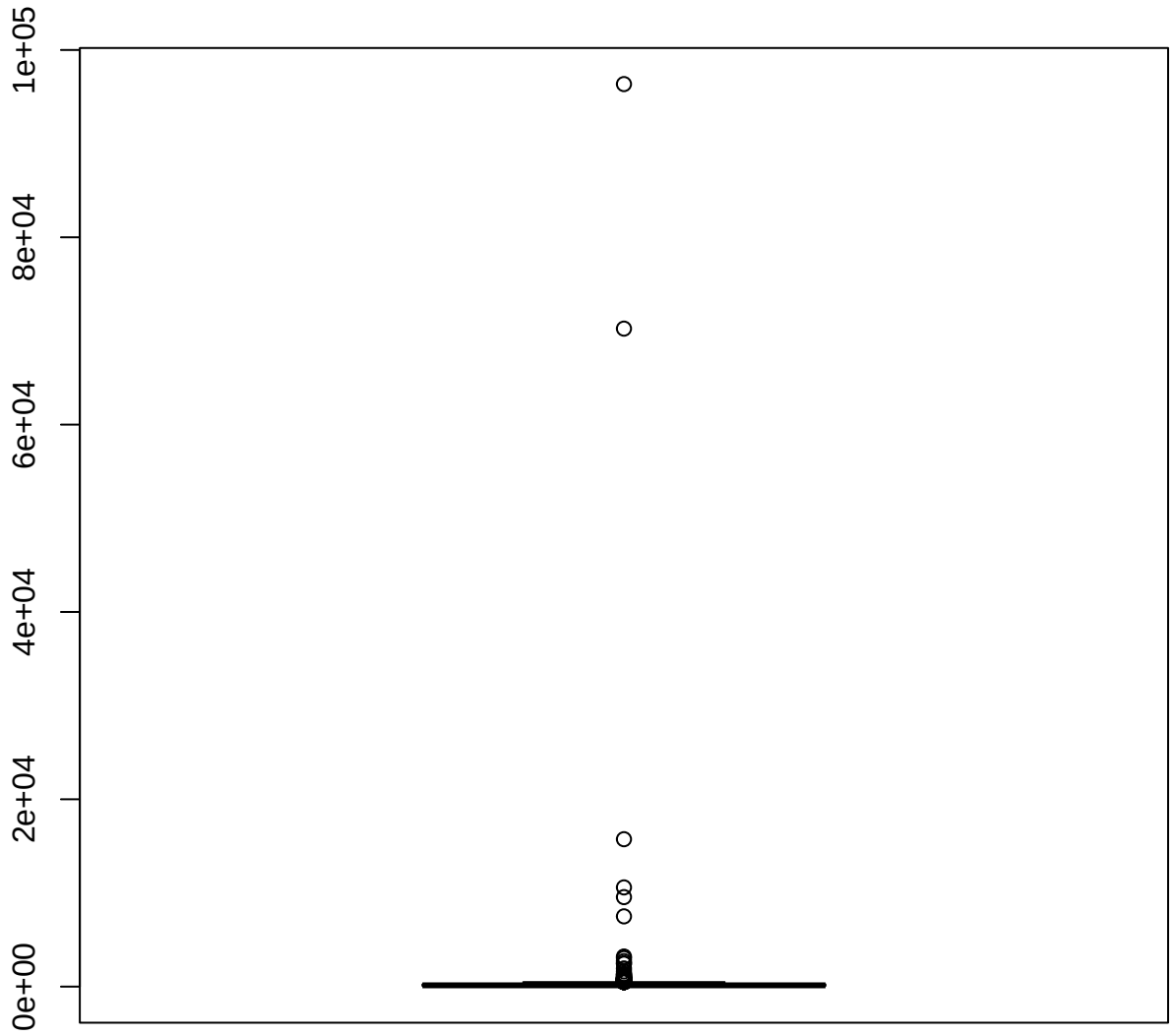
```
startOverall <- Sys.time()
library(rio)
library(dplyr)
library(MASS)
library(car)
library(QuantPsyc)
library(sandwich)
library(glmnet)
library(e1071)
library(rpart)
library(kernlab)
library(tidyverse)
library(caret)
library(xgboost)
library(class)
library(caret)
library(olsrr)
library(qpcR)
library(mosaic)
library(mosaicModel)
library(ISLR)
library(bestglm)
# library(glmulti)
library(rminer)
```

Reading and Cleaning Data

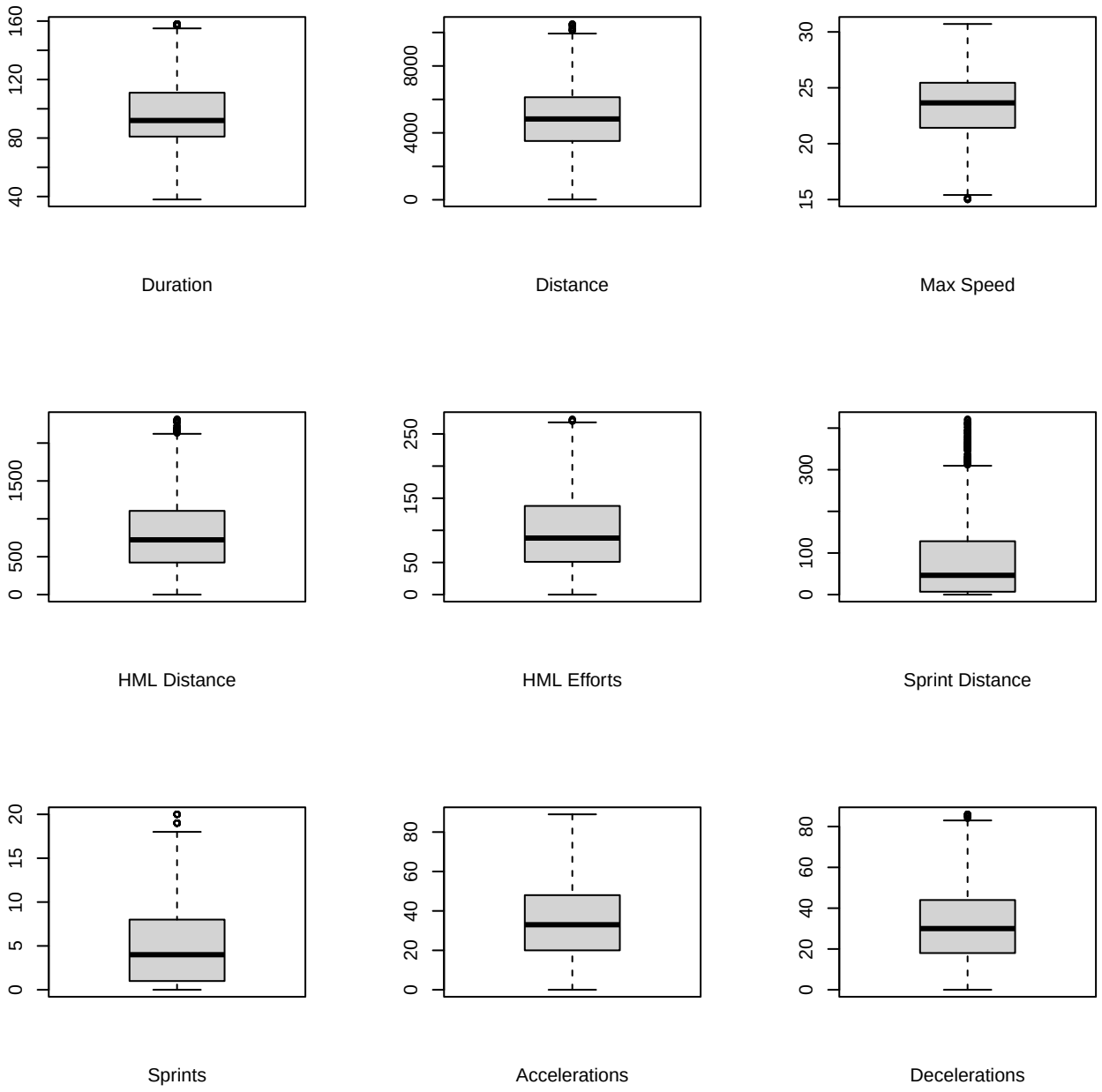
```
data1 <- import("~/Desktop/VT WSOC/Fall 2020 Recovery and RPE Scores - F19.csv")

#residuals
#t-test
#anova

allDataNoLag <- data1[c(10:28)] #all data for same day (No REC and No Lag)
allDataNoLag[] <- lapply(allDataNoLag[], function(x) round(as.numeric(x),2)) #transform '-' into NA's

allDataNoLag <- allDataNoLag %>% dplyr::rename(Distance = `Distance Total`) #2407x19

allDataNoNAs <- na.omit(allDataNoLag) #1716x19

########################################Removing Outliers

#Removing avg hr, max hr, heart rate exertion
testAllData <- allDataNoNAs[-c(11,12,15)] #1716x16


boxplot(testAllData$RPE)
```



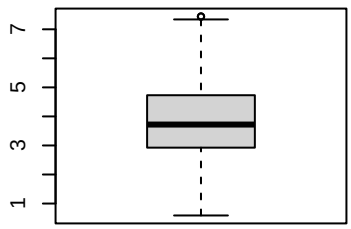

```
boxplot(testAllData$`Dynamic Stress Load`)
```



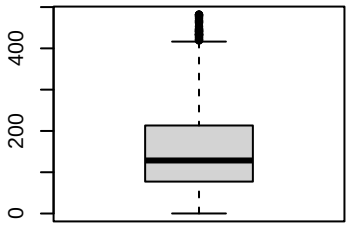

```
par(mfrow=c(3,3))
for (i in names(testAllData[c(2:16)])) {
  #print(testAllData[,1])
  Q <- quantile(testAllData[,i], probs=c(.25, .75), na.rm = FALSE)
  iqr <- IQR(testAllData[,i])
  eliminated<- subset(testAllData, testAllData[,i] > (Q[1] - 1.5*iqr)
                      & testAllData[i] < (Q[2]+1.5*iqr))

  boxplot(eliminated[,i], xlab = i)
}
```



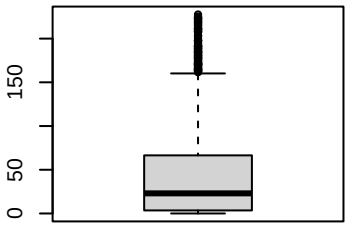

```
#eliminated = 1589x16

lmElim <- lm(RPE ~ Distance + `Max Speed` + `HML Distance` + `HML Efforts` + `Sprint Distance` + Sprints +
               Accelerations + Decelerations +  `Average Metabolic Power` + `Dynamic Stress Load` +
               `High Speed Running (Relative)` + `HML Density` + `Speed Intensity` +
               Impacts + Duration, data = eliminated)


outlierTest(lmElim) #but still has bad max speeds

      rstudent unadjusted p-value Bonferroni p
1757 -6.369954         2.4770e-10   3.9360e-07
2164  4.345073         1.4815e-05   2.3541e-02

cutoff <- 4/((nrow(eliminated)-length(lmElim$coefficients)-1))
plot(lmElim, which = 4, cook.levels = cutoff)

########################################Normality and Plotting
###External
plot(rstandard(lmElim), main = "Standardized Residual Plot")
par(mfrow=c(1,1))
```



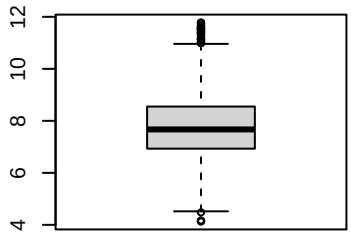
Average Metabolic Power



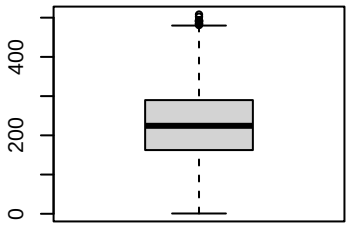
Dynamic Stress Load



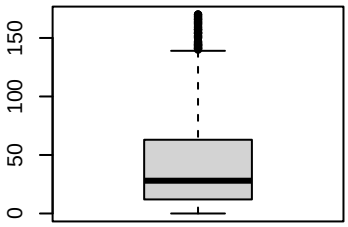
High Speed Running (Relative)



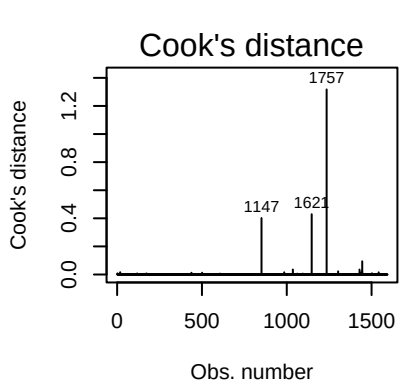
HML Density

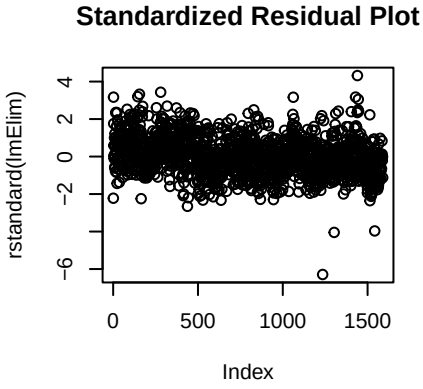


Speed Intensity

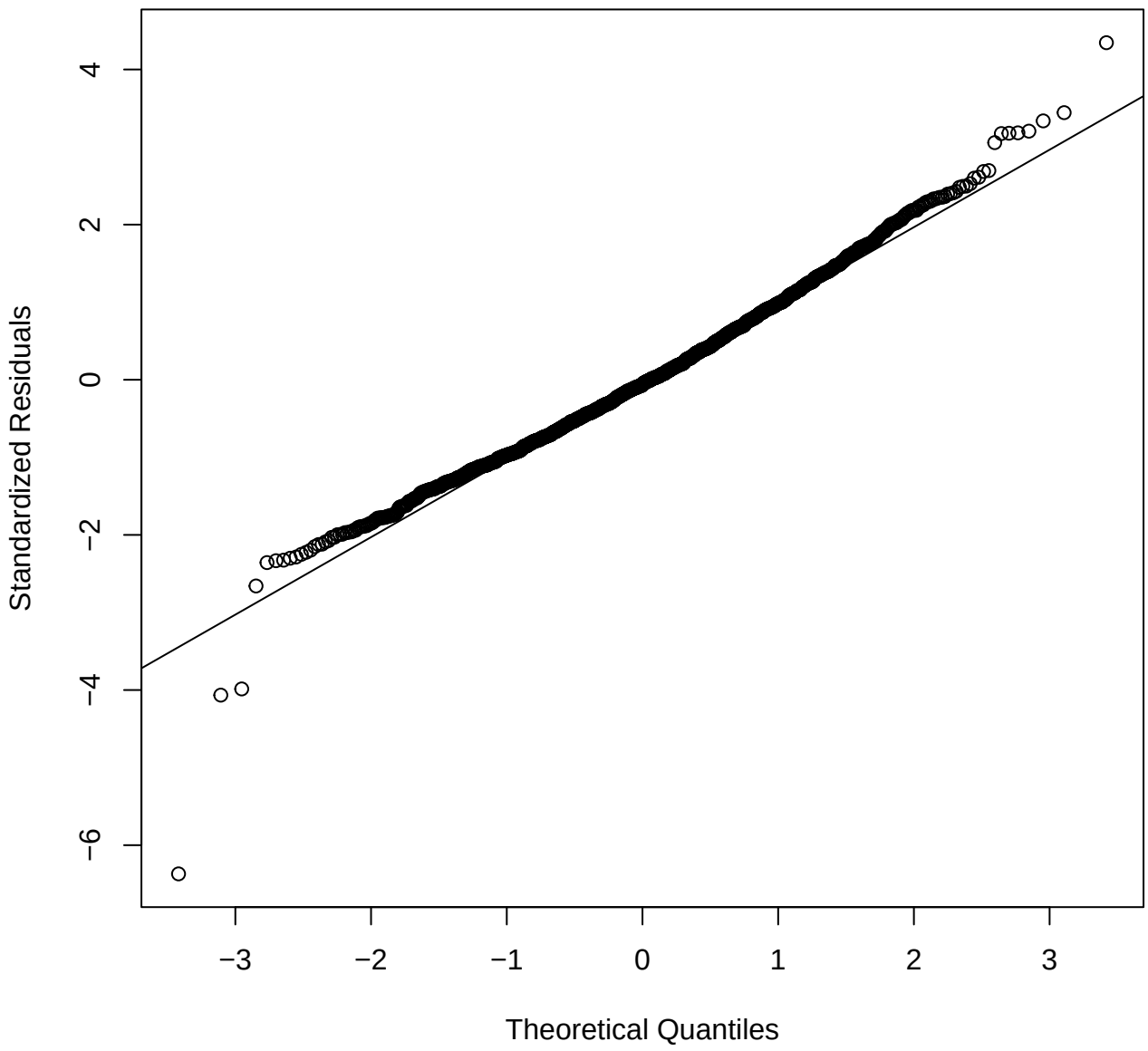


Impacts

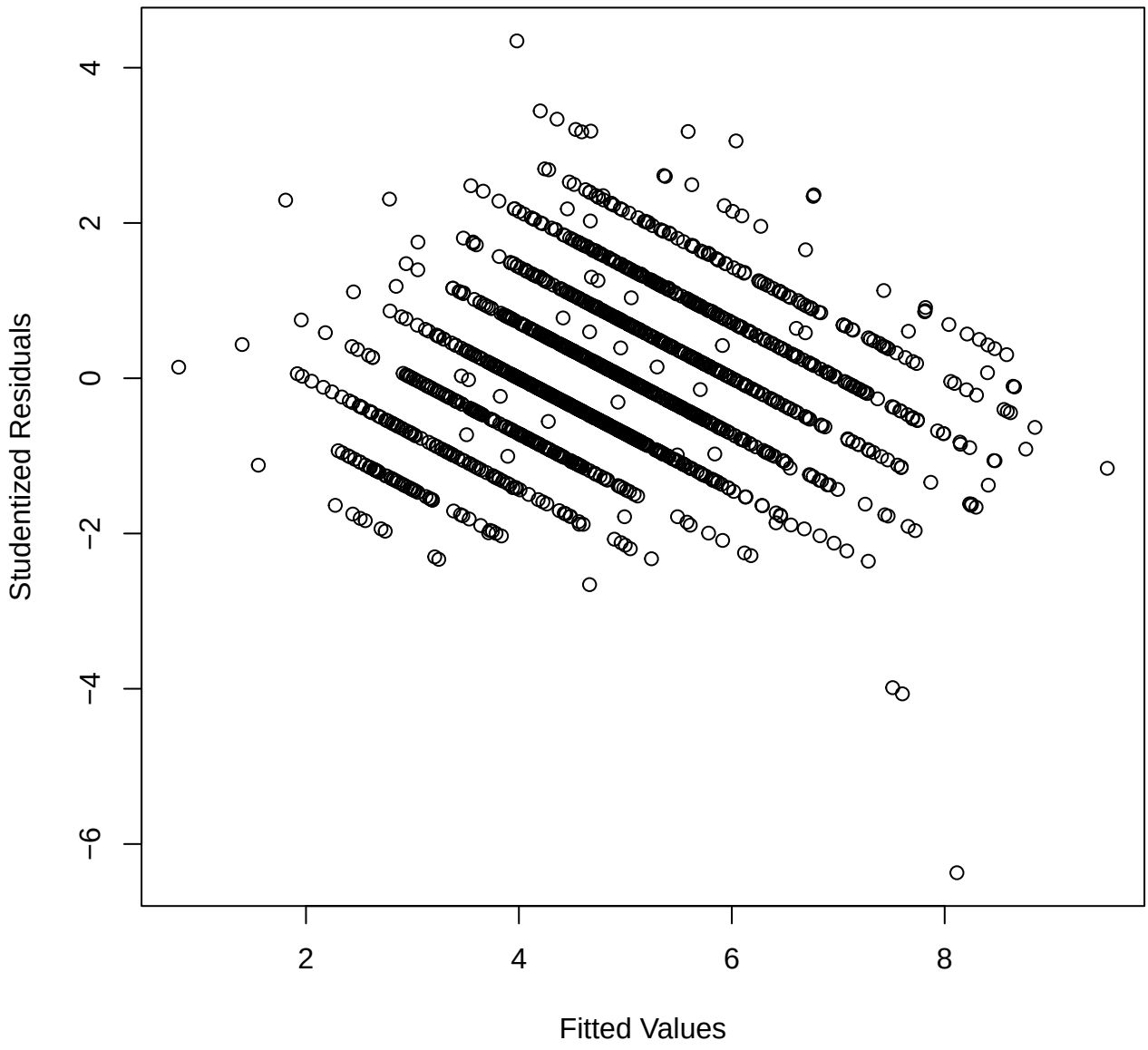




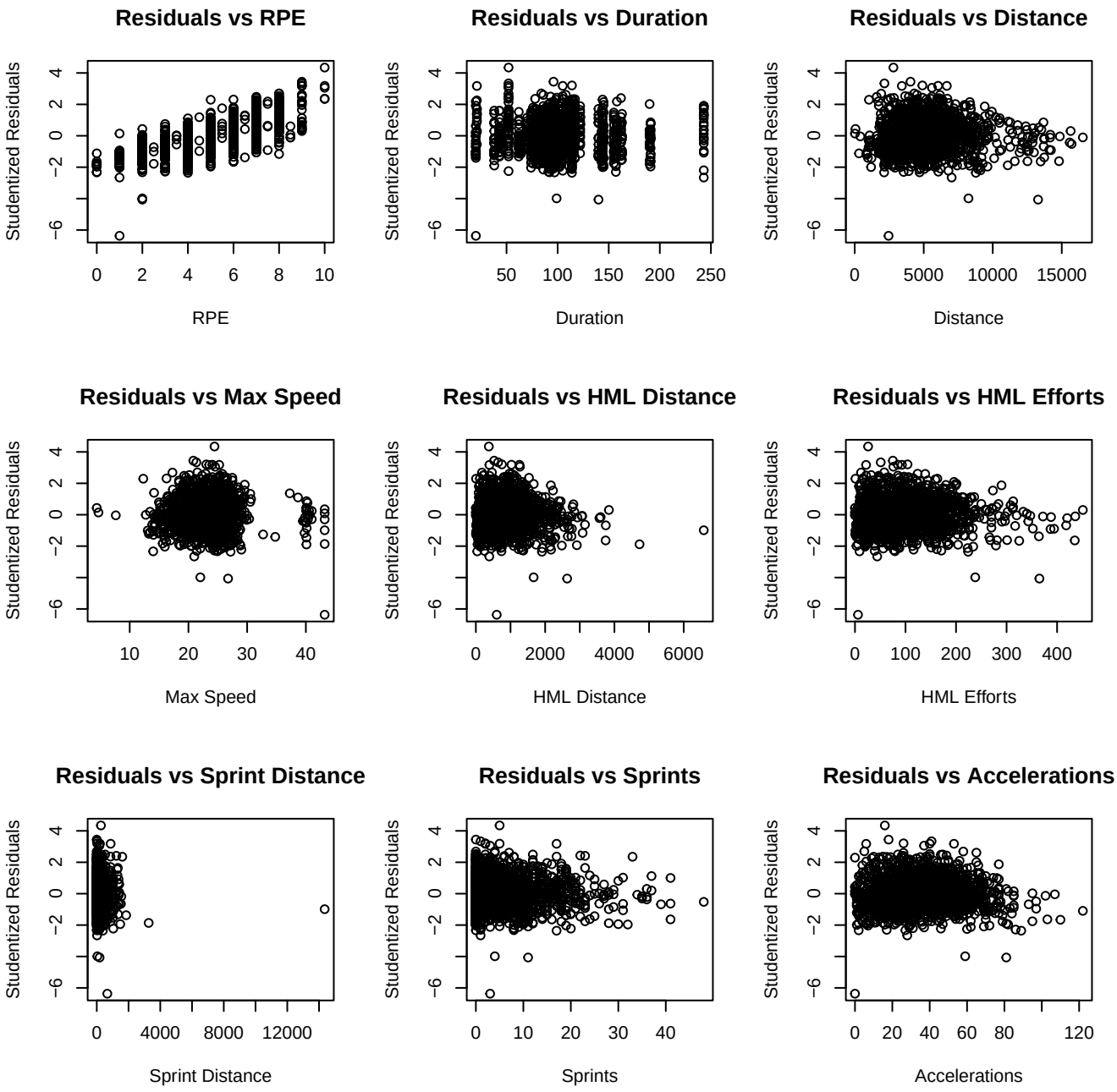

```
qqnorm(rstudent(lmElim), ylab="Standardized Residuals") #Plot normal prob with external residuals
qqline(rstudent(lmElim))
```



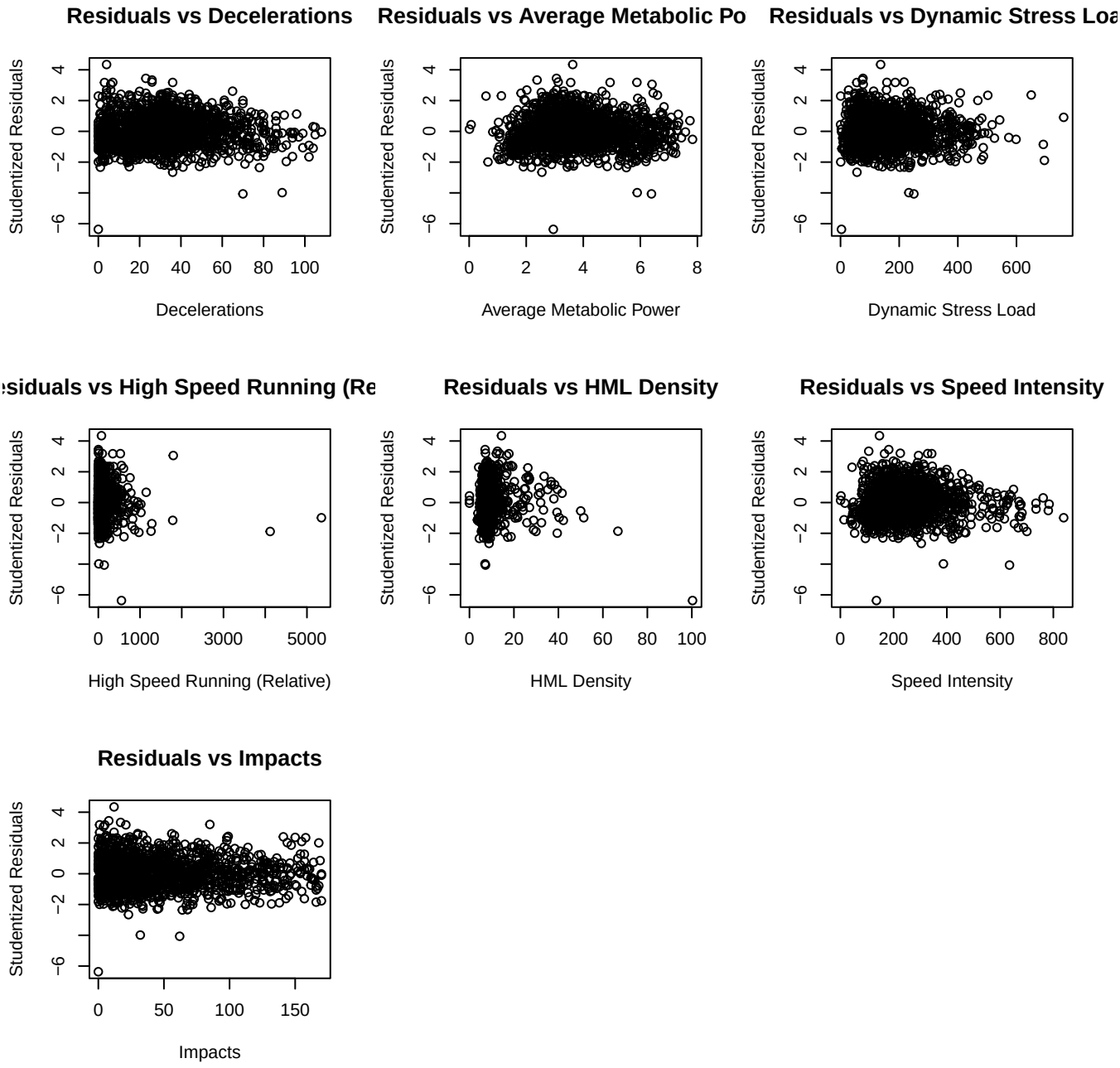

```
#External residuals vs fitted for all 19
plot(rstudent(lmElim) ~ fitted(lmElim), xlab = "Fitted Values", ylab = "Studentized Residuals", main = "Residu
```




```
#External vs each variable
par(mfrow=c(3,3))
plot(rstudent(lmElim) ~ eliminated$RPE, ylab="Studentized Residuals", xlab = "RPE", main = "Residuals vs RPE")
plot(rstudent(lmElim) ~ eliminated$Duration, ylab="Studentized Residuals", xlab = "Duration",
     main = "Residuals vs Duration")
plot(rstudent(lmElim) ~ eliminated$Distance, ylab="Studentized Residuals", xlab = "Distance",
     main = "Residuals vs Distance")
plot(rstudent(lmElim) ~ eliminated$`Max Speed`, ylab="Studentized Residuals", xlab = "Max Speed",
     main = "Residuals vs Max Speed")
plot(rstudent(lmElim) ~ eliminated$`HML Distance`, ylab="Studentized Residuals", xlab = "HML Distance",
     main = "Residuals vs HML Distance")
plot(rstudent(lmElim) ~ eliminated$`HML Efforts`, ylab="Studentized Residuals", xlab = "HML Efforts",
     main = "Residuals vs HML Efforts")
plot(rstudent(lmElim) ~ eliminated$`Sprint Distance`, ylab="Studentized Residuals", xlab = "Sprint Distance",
     main = "Residuals vs Sprint Distance")
plot(rstudent(lmElim) ~ eliminated$Sprints, ylab="Studentized Residuals", xlab = "Sprints",
     main = "Residuals vs Sprints")
plot(rstudent(lmElim) ~ eliminated$Accelerations, ylab="Studentized Residuals", xlab = "Accelerations",
     main = "Residuals vs Accelerations")
```




```
plot(rstudent(lmElim) ~ eliminated$Decelerations, ylab="Studentized Residuals", xlab = "Decelerations",
     main = "Residuals vs Decelerations")
plot(rstudent(lmElim) ~ eliminated$`Average Metabolic Power`, ylab="Studentized Residuals",
     xlab = "Average Metabolic Power", main = "Residuals vs Average Metabolic Power")
plot(rstudent(lmElim) ~ eliminated$`Dynamic Stress Load`, ylab="Studentized Residuals",
     xlab = "Dynamic Stress Load", main = "Residuals vs Dynamic Stress Load")
plot(rstudent(lmElim) ~ eliminated$`High Speed Running (Relative)`, ylab="Studentized Residuals",
     xlab = "High Speed Running (Relative)", main = "Residuals vs High Speed Running (Relative)")
plot(rstudent(lmElim) ~ eliminated$`HML Density`, ylab="Studentized Residuals", xlab = "HML Density",
     main = "Residuals vs HML Density")
plot(rstudent(lmElim) ~ eliminated$`Speed Intensity`, ylab="Studentized Residuals", xlab = "Speed Intensity",
     main = "Residuals vs Speed Intensity")
plot(rstudent(lmElim) ~ eliminated$Impacts, ylab="Studentized Residuals", xlab = "Impacts",
     main = "Residuals vs Impacts")
```



Splitting Into 3 Datasets From Model Validation

```
set.seed(42)
splits = c(train = .6, test = .2, validate = .2)

#random assigning
g = sample(cut(
  seq(nrow(eliminated)),
  nrow(eliminated)*cumsum(c(0,splits)),
  labels = names(splits)
))

res = split(eliminated, g)

sapply(res, nrow)/nrow(eliminated)
```

```
    train      test  validate
```

```
0.5997483 0.2001259 0.2001259
```

```
addmargins(prop.table(table(g)))
```

```
g
    train      test  validate       Sum
0.5997483 0.2001259 0.2001259 1.0000000
```

```
dim(res$train) #953x16
```

```
[1] 953  16
```

```
dim(res$test) #318x16
```

```
[1] 318  16
```

```
dim(res$validate) #318x16
```

```
[1] 318  16
```

```
lmTrain <- lm(RPE ~ Distance + `Max Speed` + `HML Distance` + `HML Efforts` + `Sprint Distance` + Sprints +
                Accelerations + Decelerations +  `Average Metabolic Power` + `Dynamic Stress Load` +
                `High Speed Running (Relative)` + `HML Density` + `Speed Intensity` +
                Impacts + Duration, data = res$train)

lmTest <- lm(RPE ~ Distance + `Max Speed` + `HML Distance` + `HML Efforts` + `Sprint Distance` + Sprints +
               Accelerations + Decelerations +  `Average Metabolic Power` + `Dynamic Stress Load` +
               `High Speed Running (Relative)` + `HML Density` + `Speed Intensity` +
               Impacts + Duration, data = res$test)
```

Creating Helper Functions

```
PRESS1 <- function(linear.model) {
  #' calculate the predictive residuals
  pr <- residuals(linear.model)/(1-lm.influence(linear.model)$hat)
  #' calculate the PRESS
  PRESS <- sum(pr^2)

  return(PRESS)
}

normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x))) }

get_best_result <- function(caret_fit) {
  best = which(rownames(caret_fit$results) == rownames(caret_fit$bestTune))
  best_result = caret_fit$results[best, ]
  rownames(best_result) = NULL
  best_result
}
```

All Possible Regressions

```
##All possible regressions
start_time <- Sys.time()
allRegTrain <- ols_step_all_possible(lmTrain)
end_time <- Sys.time()
time <- end_time - start_time
time #Time difference of 5.411676 mins
```

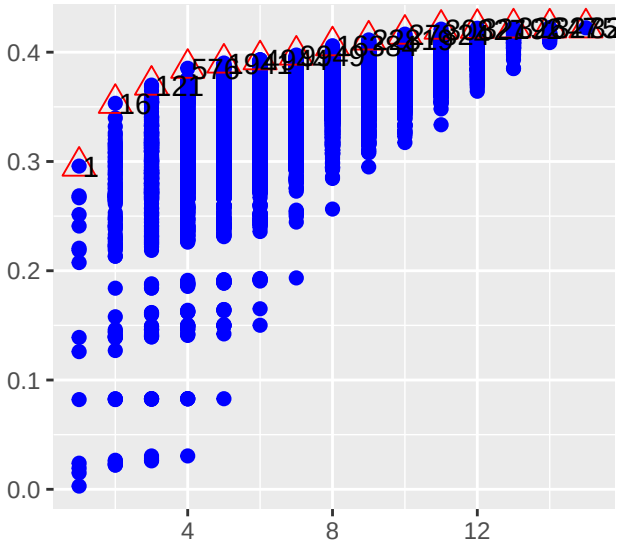
```
Time difference of 5.916364 mins
```

```
head(allRegTrain)
```

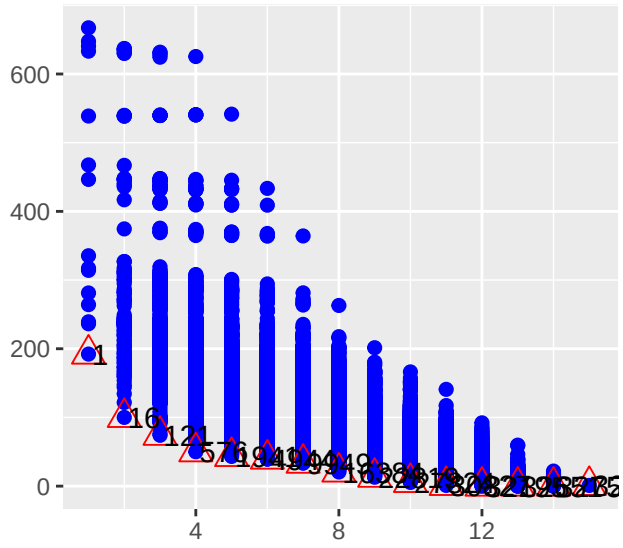
```
   Index N                Predictors  R-Square Adj. R-Square Mallow's Cp
9      1 1 `Average Metabolic Power` 0.2957601     0.2950196    193.2523
13     2 1         `Speed Intensity` 0.2685622     0.2677931    237.3664
1      3 1                  Distance 0.2668013     0.2660303    240.2226
8      4 1             Decelerations 0.2513998     0.2506126    265.2032
3      5 1            `HML Distance` 0.2409253     0.2401271    282.1924
7      6 1             Accelerations 0.2206092     0.2197896    315.1445
```

```
plot(allRegTrain)
```

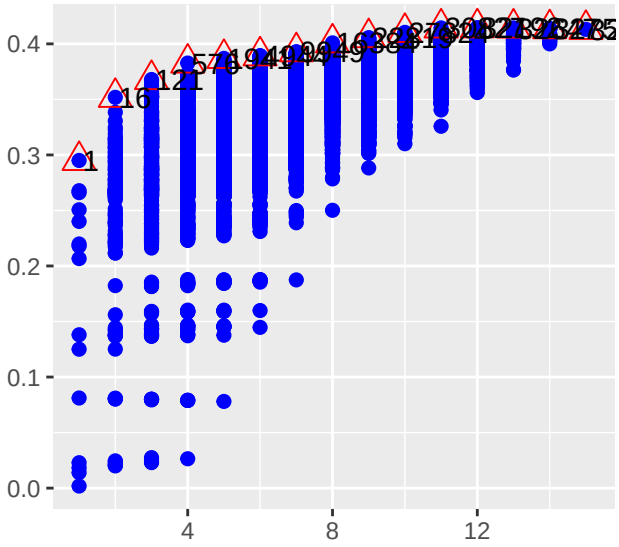
R−Square



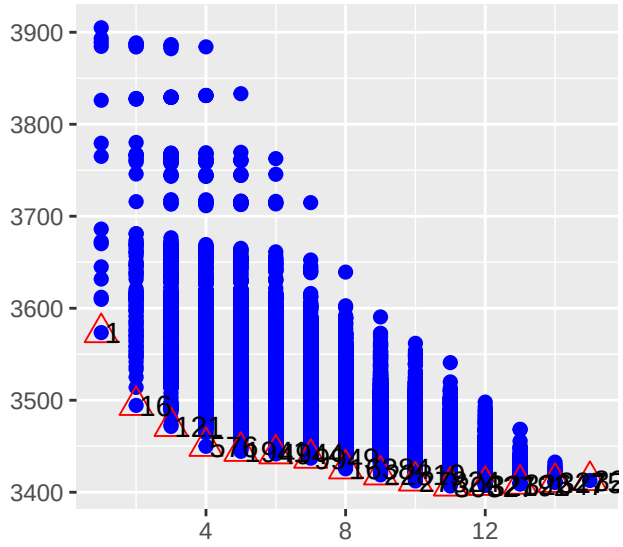
Cp

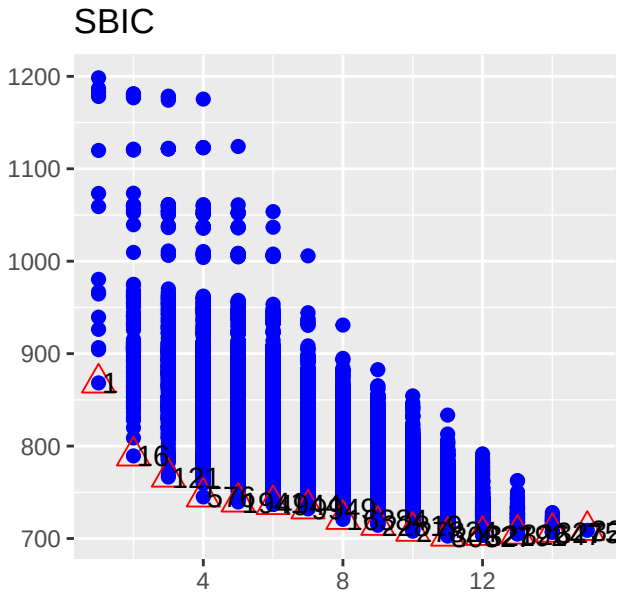


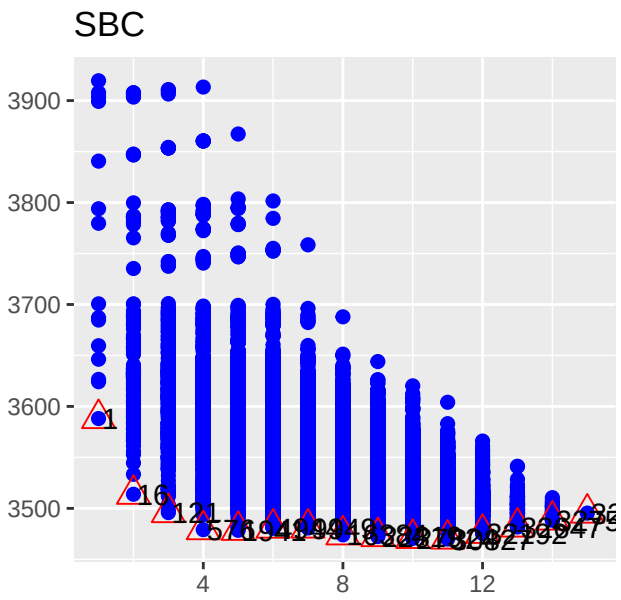
Adj. R−Square



AIC








```
allRegTrain$model
```

```
NULL
```

```
# allRegTrain[allRegTrain$mindex == 32192]
```

```
min(allRegTrain$aic[allRegTrain$n == 15])
```

```
[1] 3412.763
```

```
# s1 <- allRegTrain[allRegTrain$n == 14]
```

```
# Sort by column index [2] then [5]
sortedART <- allRegTrain[order( allRegTrain[,5], allRegTrain[,2] ),]
```

```
k11 <- lm(RPE ~ Distance + `HML Distance` + `HML Efforts` +
            Accelerations + Decelerations +  `Average Metabolic Power` + `Dynamic Stress Load` +
```

```
            `HML Density` + `Speed Intensity` + Impacts + Duration, data = res$train)
k12 <- lm(RPE ~ Distance + `HML Distance` + `HML Efforts` + Sprints +
            Accelerations + Decelerations +  `Average Metabolic Power` + `Dynamic Stress Load` +
            `HML Density` + `Speed Intensity` + Impacts + Duration, data = res$train)
PRESS1(k11) #press = 2047.351
```

```
[1] 2047.351
```

```
PRESS1(k12) #press = 2048.812
```

```
[1] 2048.812
```

```
########################################Forward AIC
start_time <- Sys.time()
forwardTrain <- ols_step_forward_aic(lmTrain)
end_time <- Sys.time()
time <- end_time - start_time
time #Time difference of 0.5972259 secs
```

```
Time difference of 0.5671031 secs
```

```
forwardTrain
```

```
                              Selection Summary
-------------------------------------------------------------------------------------
Variable                        AIC        Sum Sq       RSS        R-Sq      Adj. R-Sq
-------------------------------------------------------------------------------------
‘Average Metabolic Power‘    3573.527     989.934    2357.149    0.29576       0.29502
Decelerations                3494.414    1182.258    2164.825    0.35322       0.35186
‘HML Density‘                3471.546    1238.018    2109.065    0.36988       0.36789
‘Dynamic Stress Load‘        3449.914    1289.674    2057.409    0.38531       0.38272
Impacts                      3444.454    1305.715    2041.367    0.39011       0.38689
‘Max Speed‘                  3441.763    1315.739    2031.344    0.39310       0.38925
Accelerations                3440.713    1322.232    2024.851    0.39504       0.39056
‘HML Efforts‘                3439.720    1328.579    2018.504    0.39694       0.39183
Distance                     3435.281    1342.173    2004.910    0.40100       0.39528
Duration                     3429.807    1357.834    1989.249    0.40568       0.39937
‘Speed Intensity‘            3422.228    1377.729    1969.354    0.41162       0.40474
‘HML Distance‘               3407.718    1411.554    1935.529    0.42173       0.41434
-------------------------------------------------------------------------------------
```

```
sort(forwardTrain$model$coefficients)
```

```
        ‘Speed Intensity‘              ‘Max Speed‘             ‘HML Efforts‘
             -0.046485507              -0.015022631              -0.012455304
                  Impacts                  Duration            ‘HML Distance‘
             -0.009043414              -0.007221473               0.001540433
                 Distance     ‘Dynamic Stress Load‘             Accelerations
              0.002372524               0.006303617               0.011736891
            Decelerations            ‘HML Density‘ ‘Average Metabolic Power‘
              0.018118086               0.036612296               0.257091016
               (Intercept)
              2.076570629
```

```
########################################Backward AIC
```

```
start_time <- Sys.time()
backwardTrain <- ols_step_backward_aic(lmTrain)
end_time <- Sys.time()
time <- end_time - start_time
time #Time difference of 0.539556 secs
```

```
Time difference of 0.5383501 secs
```

```
backwardTrain
```

```
                                Backward Elimination Summary
-----------------------------------------------------------------------------------------------
Variable                              AIC         RSS        Sum Sq      R-Sq      Adj. R-Sq
-----------------------------------------------------------------------------------------------
Full Model                        3412.763    1933.591    1413.492    0.42231       0.41306
‘Sprint Distance‘                 3410.825    1933.718    1413.365    0.42227       0.41364
‘High Speed Running (Relative)‘   3408.844    1933.756    1413.327    0.42226       0.41426
‘Max Speed‘                       3407.481    1935.050    1412.033    0.42187       0.41449
Sprints                           3406.750    1937.628    1409.455    0.42110       0.41433
-----------------------------------------------------------------------------------------------
```

```
sort(backwardTrain$model$coefficients)
```

```
       ‘Speed Intensity‘             ‘HML Efforts‘                     Impacts
             -0.047916535              -0.012651098              -0.009304997
                 Duration            ‘HML Distance‘                 Distance
             -0.007453879               0.001564999               0.002434436
    ‘Dynamic Stress Load‘             Accelerations            Decelerations
              0.006345592               0.011994676               0.017780378
           ‘HML Density‘ ‘Average Metabolic Power‘               (Intercept)
              0.032775212               0.252253302               1.834814023
```

```
########################################Stepwise AIC

stepwiseTrain0 <- lm(RPE ~ 1, data = res$train) #empty modfit
stepwiseTrainAll <- lm(RPE ~ ., data = res$train) #full modfit , data = stddata?
#stddata <- scale(res$train)
#lm.beta(stepwiseTrainAll)

start_time <- Sys.time()
stepwiseTrainBest <- stepAIC(stepwiseTrainAll, direction="both",
                             scope=list(upper=stepwiseTrainAll, lower=stepwiseTrain0) ) #finds best modfit
```

```
Start:  AIC=706.27
RPE ~ Duration + Distance + ‘Max Speed‘ + ‘HML Distance‘ + ‘HML Efforts‘ +
    ‘Sprint Distance‘ + Sprints + Accelerations + Decelerations +
    ‘Average Metabolic Power‘ + ‘Dynamic Stress Load‘ + ‘High Speed Running (Relative)‘ +
    ‘HML Density‘ + ‘Speed Intensity‘ + Impacts

                                Df Sum of Sq    RSS    AIC
- ‘Sprint Distance‘               1     0.126 1933.7 704.33
- ‘High Speed Running (Relative)‘ 1     0.135 1933.7 704.33
- ‘Max Speed‘                     1     1.140 1934.7 704.83
- Sprints                         1     1.550 1935.1 705.03
<none>                                       1933.6 706.27
- Accelerations                   1    14.684 1948.3 711.48
- Impacts                         1    16.723 1950.3 712.47
- ‘HML Density‘                   1    19.194 1952.8 713.68
```

```
- Duration                          1    23.090 1956.7 715.58
- ‘Average Metabolic Power‘         1    26.623 1960.2 717.30
- Decelerations                     1    27.652 1961.2 717.80
- ‘Speed Intensity‘                 1    28.151 1961.7 718.04
- ‘HML Distance‘                    1    31.646 1965.2 719.74
- Distance                          1    34.727 1968.3 721.23
- ‘HML Efforts‘                     1    38.363 1972.0 722.99
- ‘Dynamic Stress Load‘             1    45.147 1978.7 726.26

Step:  AIC=704.33
RPE ~ Duration + Distance + ‘Max Speed‘ + ‘HML Distance‘ + ‘HML Efforts‘ +
    Sprints + Accelerations + Decelerations + ‘Average Metabolic Power‘ +
    ‘Dynamic Stress Load‘ + ‘High Speed Running (Relative)‘ +
    ‘HML Density‘ + ‘Speed Intensity‘ + Impacts

                                  Df Sum of Sq    RSS    AIC
- ‘High Speed Running (Relative)‘  1     0.038 1933.8 702.35
- ‘Max Speed‘                      1     1.206 1934.9 702.92
- Sprints                          1     1.806 1935.5 703.22
<none>                                       1933.7 704.33
+ ‘Sprint Distance‘                1     0.126 1933.6 706.27
- Accelerations                    1    14.596 1948.3 709.49
- Impacts                          1    16.597 1950.3 710.47
- ‘HML Density‘                    1    19.090 1952.8 711.69
- Duration                         1    22.970 1956.7 713.58
- ‘Average Metabolic Power‘        1    26.514 1960.2 715.31
- Decelerations                    1    28.232 1962.0 716.14
- ‘HML Distance‘                   1    31.565 1965.3 717.76
- ‘Speed Intensity‘                1    36.718 1970.4 720.25
- ‘HML Efforts‘                    1    38.496 1972.2 721.11
- ‘Dynamic Stress Load‘            1    45.022 1978.7 724.26
- Distance                         1    45.454 1979.2 724.47

Step:  AIC=702.35
RPE ~ Duration + Distance + ‘Max Speed‘ + ‘HML Distance‘ + ‘HML Efforts‘ +
    Sprints + Accelerations + Decelerations + ‘Average Metabolic Power‘ +
    ‘Dynamic Stress Load‘ + ‘HML Density‘ + ‘Speed Intensity‘ +
    Impacts

                                  Df Sum of Sq    RSS    AIC
- ‘Max Speed‘                      1     1.294 1935.0 700.98
- Sprints                          1     1.774 1935.5 701.22
<none>                                       1933.8 702.35
+ ‘High Speed Running (Relative)‘  1     0.038 1933.7 704.33
+ ‘Sprint Distance‘                1     0.030 1933.7 704.33
- Accelerations                    1    14.759 1948.5 707.59
- Impacts                          1    17.397 1951.2 708.88
- ‘HML Density‘                    1    19.135 1952.9 709.73
- Duration                         1    23.225 1957.0 711.72
- ‘Average Metabolic Power‘        1    27.437 1961.2 713.77
- Decelerations                    1    28.261 1962.0 714.17
- ‘HML Distance‘                   1    35.214 1969.0 717.55
- ‘HML Efforts‘                    1    47.037 1980.8 723.25
- ‘Speed Intensity‘                1    48.140 1981.9 723.78
- ‘Dynamic Stress Load‘            1    49.219 1983.0 724.30
- Distance                         1    57.733 1991.5 728.38

Step:  AIC=700.98
RPE ~ Duration + Distance + ‘HML Distance‘ + ‘HML Efforts‘ +
    Sprints + Accelerations + Decelerations + ‘Average Metabolic Power‘ +
```

```
    `Dynamic Stress Load` + `HML Density` + `Speed Intensity` +
    Impacts

                                Df Sum of Sq    RSS    AIC
- Sprints                        1     2.579 1937.6 700.25
<none>                                       1935.0 700.98
+ `Max Speed`                   1     1.294 1933.8 702.35
+ `High Speed Running (Relative)` 1     0.126 1934.9 702.92
+ `Sprint Distance`             1     0.021 1935.0 702.97
- Accelerations                 1    15.347 1950.4 706.51
- `HML Density`                 1    17.937 1953.0 707.78
- Impacts                       1    18.114 1953.2 707.86
- Duration                      1    24.419 1959.5 710.94
- `Average Metabolic Power`     1    26.997 1962.0 712.19
- Decelerations                 1    27.634 1962.7 712.50
- `HML Distance`                1    36.685 1971.7 716.88
- `HML Efforts`                 1    48.935 1984.0 722.78
- `Dynamic Stress Load`         1    49.921 1985.0 723.26
- `Speed Intensity`             1    50.514 1985.6 723.54
- Distance                      1    60.074 1995.1 728.12

Step:  AIC=700.25
RPE ~ Duration + Distance + `HML Distance` + `HML Efforts` +
    Accelerations + Decelerations + `Average Metabolic Power` +
    `Dynamic Stress Load` + `HML Density` + `Speed Intensity` +
    Impacts

                                Df Sum of Sq    RSS    AIC
<none>                                       1937.6 700.25
+ Sprints                       1     2.579 1935.0 700.98
+ `Max Speed`                   1     2.099 1935.5 701.22
+ `Sprint Distance`             1     0.349 1937.3 702.08
+ `High Speed Running (Relative)` 1     0.005 1937.6 702.25
- Accelerations                 1    15.285 1952.9 705.74
- `HML Density`                 1    17.786 1955.4 706.96
- Impacts                       1    19.660 1957.3 707.87
- `Average Metabolic Power`     1    25.352 1963.0 710.64
- Duration                      1    26.369 1964.0 711.14
- Decelerations                 1    26.987 1964.6 711.44
- `HML Distance`                1    35.056 1972.7 715.34
- `HML Efforts`                 1    47.334 1985.0 721.25
- `Dynamic Stress Load`         1    49.297 1986.9 722.20
- `Speed Intensity`             1    58.214 1995.8 726.46
- Distance                      1    68.354 2006.0 731.29

end_time <- Sys.time()
time <- end_time - start_time
time #Time difference of 0.4075069 secs

Time difference of 0.05228782 secs

formula(stepwiseTrainBest)

RPE ~ Duration + Distance + `HML Distance` + `HML Efforts` +
    Accelerations + Decelerations + `Average Metabolic Power` +
    `Dynamic Stress Load` + `HML Density` + `Speed Intensity` +
    Impacts

summary(stepwiseTrainBest)
```

```
Call:
lm(formula = RPE ~ Duration + Distance + `HML Distance` + `HML Efforts` +
    Accelerations + Decelerations + `Average Metabolic Power` +
    `Dynamic Stress Load` + `HML Density` + `Speed Intensity` +
    Impacts, data = res$train)

Residuals:
    Min      1Q  Median      3Q     Max
-5.3919 -0.9847 -0.0819  0.9836  6.0939

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)               1.8348140  0.2853966   6.429 2.04e-10 ***
Duration                 -0.0074539  0.0020830  -3.579 0.000363 ***
Distance                  0.0024344  0.0004225   5.762 1.13e-08 ***
`HML Distance`            0.0015650  0.0003793   4.126 4.02e-05 ***
`HML Efforts`            -0.0126511  0.0026386  -4.795 1.89e-06 ***
Accelerations             0.0119947  0.0044025   2.725 0.006559 **
Decelerations             0.0177804  0.0049114   3.620 0.000310 ***
`Average Metabolic Power` 0.2522533  0.0718901   3.509 0.000471 ***
`Dynamic Stress Load`     0.0063456  0.0012969   4.893 1.17e-06 ***
`HML Density`             0.0327752  0.0111519   2.939 0.003373 **
`Speed Intensity`        -0.0479165  0.0090118  -5.317 1.32e-07 ***
Impacts                  -0.0093050  0.0030114  -3.090 0.002061 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.435 on 941 degrees of freedom
Multiple R-squared:  0.4211,    Adjusted R-squared:  0.4143
F-statistic: 62.23 on 11 and 941 DF,  p-value: < 2.2e-16

sort(coefficients(stepwiseTrainBest))

       `Speed Intensity`            `HML Efforts`                  Impacts
             -0.047916535             -0.012651098             -0.009304997
                 Duration           `HML Distance`                 Distance
             -0.007453879              0.001564999              0.002434436
   `Dynamic Stress Load`             Accelerations            Decelerations
              0.006345592              0.011994676              0.017780378
          `HML Density` `Average Metabolic Power`              (Intercept)
              0.032775212              0.252253302              1.834814023

Rsq.ad(stepwiseTrainBest)

[1] 0.4143322

AIC(stepwiseTrainBest)

[1] 3406.75

########################################Normal GLM
start_time <- Sys.time()
summary(glmTrain <- glm(RPE ~ Distance + `Max Speed` + `HML Distance` + `HML Efforts` +
                          `Sprint Distance` + Sprints + Accelerations + Decelerations +
                          `Average Metabolic Power` + `Dynamic Stress Load` +
                          `High Speed Running (Relative)` + `HML Density` +
                          `Speed Intensity` + Impacts + Duration, data = res$train))
```

```
Call:
glm(formula = RPE ~ Distance + `Max Speed` + `HML Distance` +
    `HML Efforts` + `Sprint Distance` + Sprints + Accelerations +
    Decelerations + `Average Metabolic Power` + `Dynamic Stress Load` +
    `High Speed Running (Relative)` + `HML Density` + `Speed Intensity` +
    Impacts + Duration, data = res$train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-5.4908  -1.0078  -0.0881   0.9608   6.1054

Coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                     1.9705187  0.3900933   5.051 5.27e-07 ***
Distance                        0.0023458  0.0005718   4.102 4.45e-05 ***
`Max Speed`                    -0.0114720  0.0154362  -0.743 0.457552
`HML Distance`                  0.0016134  0.0004120   3.916 9.66e-05 ***
`HML Efforts`                  -0.0128821  0.0029877  -4.312 1.79e-05 ***
`Sprint Distance`               0.0000664  0.0002682   0.248 0.804515
Sprints                        -0.0100572  0.0116053  -0.867 0.386381
Accelerations                   0.0118140  0.0044289   2.668 0.007774 **
Decelerations                   0.0181887  0.0049688   3.661 0.000266 ***
`Average Metabolic Power`       0.2649113  0.0737532   3.592 0.000345 ***
`Dynamic Stress Load`           0.0063158  0.0013503   4.677 3.33e-06 ***
`High Speed Running (Relative)` -0.0001464  0.0005722  -0.256 0.798173
`HML Density`                   0.0360813  0.0118306   3.050 0.002354 **
`Speed Intensity`              -0.0458772  0.0124213  -3.693 0.000234 ***
Impacts                        -0.0088076  0.0030939  -2.847 0.004513 **
Duration                       -0.0070711  0.0021139  -3.345 0.000855 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 2.063598)

    Null deviance: 3347.1  on 952  degrees of freedom
Residual deviance: 1933.6  on 937  degrees of freedom
AIC: 3412.8

Number of Fisher Scoring iterations: 2

end_time <- Sys.time()
time <- end_time - start_time
time #Time difference of 0.4860902 secs

Time difference of 0.005637884 secs

glmTrain$formula

RPE ~ Distance + `Max Speed` + `HML Distance` + `HML Efforts` +
    `Sprint Distance` + Sprints + Accelerations + Decelerations +
    `Average Metabolic Power` + `Dynamic Stress Load` + `High Speed Running (Relative)` +
    `HML Density` + `Speed Intensity` + Impacts + Duration

sort(glmTrain$coefficients) #removed max speed, sprint distance, sprints, high speed running,

              `Speed Intensity`                   `HML Efforts`
                  -0.0458772138                   -0.0128820680
                    `Max Speed`                         Sprints
```

```
                -0.0114720433                 -0.0100572421
                      Impacts                      Duration
                -0.0088075776                 -0.0070711098
`High Speed Running (Relative)`               `Sprint Distance`
                -0.0001463511                  0.0000663996
               `HML Distance`                      Distance
                 0.0016133668                  0.0023458395
          `Dynamic Stress Load`                 Accelerations
                 0.0063158154                  0.0118140359
                Decelerations                 `HML Density`
                 0.0181887254                  0.0360812768
      `Average Metabolic Power`                   (Intercept)
                 0.2649112686                  1.9705186694
```

```
Rsq.ad(glmTrain)
```

```
[1] 0.4130576
```

```
AIC(glmTrain)
```

```
[1] 3412.763
```

```
#bestglm::bestglm(res$train[,-1], IC = "AIC")

#Add the deviance residuals versus each regressor
#and normal probability plot of the residuals.

########################################Poisson Regression
start_time <- Sys.time()
summary(poissonTrain <- glm(RPE ~ Distance + `Max Speed` + `HML Distance` + `HML Efforts` + `Sprint Distance` +
                            Accelerations + Decelerations +  `Average Metabolic Power` + `Dynamic Stress Load` +
                            `High Speed Running (Relative)` + `HML Density` + `Speed Intensity` +
                            Impacts + Duration, family = "poisson", data = res$train))
```

```
Call:
glm(formula = RPE ~ Distance + `Max Speed` + `HML Distance` +
    `HML Efforts` + `Sprint Distance` + Sprints + Accelerations +
    Decelerations + `Average Metabolic Power` + `Dynamic Stress Load` +
    `High Speed Running (Relative)` + `HML Density` + `Speed Intensity` +
    Impacts + Duration, family = "poisson", data = res$train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6591  -0.4765  -0.0372   0.4315   2.5898

Coefficients:
                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                  9.315e-01  1.270e-01   7.337 2.19e-13 ***
Distance                     5.499e-04  1.756e-04   3.132  0.00174 **
`Max Speed`                  1.675e-03  4.849e-03   0.346  0.72968
`HML Distance`               3.093e-04  1.148e-04   2.695  0.00705 **
`HML Efforts`               -2.499e-03  8.421e-04  -2.967  0.00301 **
`Sprint Distance`            2.842e-05  8.056e-05   0.353  0.72421
Sprints                     -2.050e-03  3.331e-03  -0.615  0.53833
Accelerations                3.039e-03  1.343e-03   2.262  0.02367 *
Decelerations                3.059e-03  1.478e-03   2.070  0.03843 *
`Average Metabolic Power`    5.167e-02  2.278e-02   2.268  0.02332 *
`Dynamic Stress Load`        1.159e-03  3.756e-04   3.086  0.00203 **
```

```
‘High Speed Running (Relative)‘  1.280e-05  1.667e-04   0.077  0.93880
‘HML Density‘                    6.601e-03  3.474e-03   1.900  0.05742 .
‘Speed Intensity‘               -1.084e-02  3.788e-03  -2.862  0.00421 **
Impacts                         -1.538e-03  8.786e-04  -1.750  0.08007 .
Duration                        -1.947e-03  7.249e-04  -2.687  0.00722 **
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 766.47  on 952  degrees of freedom
Residual deviance: 483.34  on 937  degrees of freedom
AIC: Inf

Number of Fisher Scoring iterations: 5

end_time <- Sys.time()
time <- end_time - start_time
time #Time difference of 0.3096309 secs

Time difference of 0.006983995 secs

sort(poissonTrain$coefficients) #removed max speed, sprint distance, sprints, high speed running, HMLDe, impac

              ‘Speed Intensity‘                   ‘HML Efforts‘
                  -1.084108e-02                   -2.498541e-03
                        Sprints                        Duration
                  -2.049589e-03                   -1.947454e-03
                        Impacts ‘High Speed Running (Relative)‘
                  -1.537835e-03                    1.279919e-05
              ‘Sprint Distance‘                  ‘HML Distance‘
                   2.842357e-05                    3.093053e-04
                       Distance           ‘Dynamic Stress Load‘
                   5.499109e-04                    1.158974e-03
                    ‘Max Speed‘                   Accelerations
                   1.675493e-03                    3.038635e-03
                  Decelerations                   ‘HML Density‘
                   3.059245e-03                    6.600955e-03
      ‘Average Metabolic Power‘                     (Intercept)
                   5.166588e-02                    9.314717e-01

Rsq.ad(poissonTrain)

[1] 0.8022394

AIC(poissonTrain)

[1] Inf

?optim
?Multinomial

########################################Penalized Regression

xTrain <- model.matrix(RPE~., res$train)[,-1]
yTrain <- res$train$RPE
start_time <- Sys.time()
penReg <- glmnet(xTrain, yTrain, alpha = 1, lambda = NULL)
penReg$a0
```

```
      s0       s1       s2       s3       s4       s5       s6       s7
4.892970 4.636296 4.402424 4.186107 3.987718 3.810710 3.649592 3.503484
      s8       s9      s10      s11      s12      s13      s14      s15
3.371603 3.249603 3.133339 3.028497 2.931777 2.844285 2.764436 2.691664
     s16      s17      s18      s19      s20      s21      s22      s23
2.627941 2.549083 2.455174 2.382648 2.316199 2.255538 2.200206 2.149923
     s24      s25      s26      s27      s28      s29      s30      s31
2.104048 2.062241 2.024147 1.989437 1.987523 2.012869 2.036218 2.057509
     s32      s33      s34      s35      s36      s37      s38      s39
2.076498 2.094174 2.104928 2.130029 2.167183 2.187353 2.205855 2.222918
     s40      s41      s42      s43      s44      s45      s46      s47
2.227593 2.225113 2.213949 2.208059 2.202626 2.198090 2.201766 2.204782
     s48      s49      s50      s51      s52      s53      s54      s55
2.207416 2.209921 2.212210 2.214298 2.216205 2.217843 2.219427 2.220880
     s56      s57      s58      s59      s60      s61      s62      s63
2.222121 2.223324 2.224436 2.225385 2.226294 2.227092 2.227889 2.228557
     s64      s65      s66      s67      s68      s69      s70      s71
2.229281 2.218150 2.198536 2.180644 2.164325 2.149466 2.135928 2.123613
     s72      s73      s74      s75      s76      s77      s78      s79
2.112368 2.102121 2.092800 2.084304 2.076574 2.069519 2.063060 2.057215
     s80      s81      s82      s83      s84
2.051868 2.047010 2.042596 2.038556 2.038558
```

```
end_time <- Sys.time()
time <- end_time - start_time
time #Time difference of 0.5621359 secs
```

```
Time difference of 0.01923704 secs
```

```
########################################Ridge
start_time <- Sys.time()
cvTrain <- cv.glmnet(xTrain, yTrain, alpha = 0)
cvTrain
```

```
Call:  cv.glmnet(x = xTrain, y = yTrain, alpha = 0)

Measure: Mean-Squared Error

    Lambda Index Measure     SE Nonzero
min 0.2145    92   2.181 0.1328      15
1se 2.6448    65   2.310 0.1137      15
```

```
end_time <- Sys.time()
time <- end_time - start_time
time #Time difference of 0.375262 secs
```

```
Time difference of 0.07357502 secs
```

```
# Display the best lambda value
cvTrain$lambda.min
```

```
[1] 0.2145304
```

```
start_time <- Sys.time()
# Fit the final model on the training data
ridgeTrain <- glmnet(xTrain, yTrain, alpha = 0, lambda = cvTrain$lambda.min)
ridgeTrain$a0
```

```
      s0
2.364968
```

```
end_time <- Sys.time()
time <- end_time - start_time
time #Time difference of 0.424089 secs
```

```
Time difference of 0.001779795 secs
```

```
# Display regression coefficients
coef(ridgeTrain)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                                          s0
(Intercept)                        2.364968e+00
Duration                          -5.024759e-03
Distance                           1.022291e-04
‘Max Speed‘                       -1.239448e-02
‘HML Distance‘                     1.922094e-04
‘HML Efforts‘                     -2.097928e-03
‘Sprint Distance‘                 -2.078236e-04
Sprints                           -9.556996e-03
Accelerations                      1.191649e-02
Decelerations                      1.534253e-02
‘Average Metabolic Power‘          2.285612e-01
‘Dynamic Stress Load‘              2.835731e-03
‘High Speed Running (Relative)‘ -1.097478e-05
‘HML Density‘                      4.076874e-02
‘Speed Intensity‘                  1.544405e-03
Impacts                           -7.808974e-04
```

```
ridgeTrain
```

```
Call:  glmnet(x = xTrain, y = yTrain, alpha = 0, lambda = cvTrain$lambda.min)

  Df  %Dev Lambda
1 15 40.29 0.2145
```

```
tLLTrain <- ridgeTrain$nulldev - deviance(ridgeTrain)
kTrain <- ridgeTrain$df
nTrain <- ridgeTrain$nobs
AICcTrain <- -tLLTrain+2*kTrain+2*kTrain*(kTrain+1)/(nTrain-kTrain-1)
AICcTrain
```

```
[1] -1318.04
```

```
Rsq.ad(ridgeTrain)
```

```
[1] NaN
```

```
rsquared(ridgeTrain)
```

```
NULL
```

```
adjr2 <- ridgeTrain$dev.ratio[which(ridgeTrain$lambda == ridgeTrain$lambda)]
adjr2
```

```
[1] 0.4029039
```

```
# #change df here - test not called yet
# # Make predictions on the test data
# x.test <- model.matrix(RPE ~., test.data)[,-1]
# predictions <- model %>% predict(x.test) %>% as.vector()
# # Model performance metrics
# data.frame(
#   RMSE = RMSE(predictions, test.data$RPE),
#   Rsquare = R2(predictions, test.data$RPE)
# )

########################################Lasso

cvTrain <- cv.glmnet(xTrain, yTrain, alpha = 1)
# Display the best lambda value
cvTrain$lambda.min
```

```
[1] 0.0004114501
```

```
# Fit the final model on the training data
lassoTrain <- glmnet(xTrain, yTrain, alpha = 1, lambda = cvTrain$lambda.min)
# Dsiplay regression coefficients
coef(lassoTrain)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                                       s0
(Intercept)                     2.032925e+00
Duration                       -6.713676e-03
Distance                        1.832285e-03
`Max Speed`                    -1.240548e-02
`HML Distance`                  1.414105e-03
`HML Efforts`                  -1.209982e-02
`Sprint Distance`              -5.820908e-05
Sprints                        -1.296858e-02
Accelerations                   1.162221e-02
Decelerations                   1.897046e-02
`Average Metabolic Power`       2.580875e-01
`Dynamic Stress Load`           6.033027e-03
`High Speed Running (Relative)` -1.282020e-04
`HML Density`                   3.678165e-02
`Speed Intensity`              -3.470658e-02
Impacts                        -8.162990e-03
```

```
tLLTrainLasso <- lassoTrain$nulldev - deviance(lassoTrain)
kTrainLasso <- lassoTrain$df
nTrainLasso <- lassoTrain$nobs
AICcTrainLasso <- -tLLTrainLasso+2*kTrainLasso+2*kTrainLasso*(kTrainLasso+1)/(nTrainLasso-kTrainLasso-1)
AICcTrainLasso
```

```
[1] -1381.284
```

```
adjr2lasso <- lassoTrain$dev.ratio[which(lassoTrain$lambda == lassoTrain$lambda)]
adjr2lasso
```

```
[1] 0.421799
```

```r
# Make predictions on the test data
# x.test <- model.matrix(RPE ~., test.data)[,-1]
# predictions <- model %>% predict(x.test) %>% as.vector()
# # Model performance metrics
# data.frame(
#   RMSE = RMSE(predictions, test.data$RPE),
#   Rsquare = R2(predictions, test.data$RPE)
# )

########################################Elastic Net

# Build the model using the training set
set.seed(42)
elasticTrain <- train(
  RPE ~., data = res$train, method = "glmnet",
  trControl = trainControl("cv", number = 5),
  tuneLength = 10
)
# Best tuning parameter
elasticTrain$bestTune
```

```
  alpha    lambda
9   0.1 0.3819565
```

```r
elasticTrain
```

```
glmnet

953 samples
 15 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 761, 763, 762, 763, 763
Resampling results across tuning parameters:

  alpha  lambda        RMSE      Rsquared   MAE
  0.1    0.0004708938  1.504459  0.3698215  1.168055
  0.1    0.0010878257  1.504424  0.3698506  1.168101
  0.1    0.0025130186  1.504811  0.3695075  1.169690
  0.1    0.0058053988  1.504685  0.3695629  1.170733
  0.1    0.0134112242  1.501454  0.3717754  1.169841
  0.1    0.0309816674  1.497263  0.3742813  1.168319
  0.1    0.0715716700  1.494025  0.3750688  1.170798
  0.1    0.1653398405  1.493614  0.3726107  1.177107
  0.1    0.3819564761  1.491124  0.3738442  1.183919
  0.1    0.8823689996  1.505814  0.3685936  1.201285
  0.2    0.0004708938  1.504706  0.3696917  1.167304
  0.2    0.0010878257  1.504717  0.3696629  1.168214
  0.2    0.0025130186  1.504963  0.3693538  1.169930
  0.2    0.0058053988  1.504623  0.3695482  1.170954
  0.2    0.0134112242  1.499987  0.3726854  1.169187
  0.2    0.0309816674  1.496478  0.3743979  1.168413
  0.2    0.0715716700  1.497915  0.3709226  1.174620
  0.2    0.1653398405  1.492574  0.3727114  1.179213
  0.2    0.3819564761  1.495970  0.3716588  1.189793
  0.2    0.8823689996  1.526325  0.3619015  1.220512
  0.3    0.0004708938  1.505155  0.3693634  1.167542
  0.3    0.0010878257  1.504992  0.3694800  1.168349
```

0.3	0.0025130186	1.505182	0.3691471	1.170237
0.3	0.0058053988	1.504190	0.3697739	1.170992
0.3	0.0134112242	1.499447	0.3730060	1.168964
0.3	0.0309816674	1.496872	0.3737036	1.170174
0.3	0.0715716700	1.497405	0.3705549	1.176052
0.3	0.1653398405	1.492317	0.3726775	1.181265
0.3	0.3819564761	1.503944	0.3674894	1.196346
0.3	0.8823689996	1.544374	0.3631737	1.241312
0.4	0.0004708938	1.505502	0.3691257	1.167707
0.4	0.0010878257	1.505322	0.3692524	1.168524
0.4	0.0025130186	1.505528	0.3688426	1.170661
0.4	0.0058053988	1.503667	0.3701107	1.170828
0.4	0.0134112242	1.499256	0.3729731	1.168991
0.4	0.0309816674	1.498571	0.3719520	1.172316
0.4	0.0715716700	1.496873	0.3703746	1.176739
0.4	0.1653398405	1.494108	0.3714168	1.183440
0.4	0.3819564761	1.511878	0.3643231	1.205708
0.4	0.8823689996	1.568913	0.3626550	1.266888
0.5	0.0004708938	1.505766	0.3689706	1.167767
0.5	0.0010878257	1.505684	0.3690087	1.168717
0.5	0.0025130186	1.506004	0.3684455	1.171179
0.5	0.0058053988	1.502967	0.3705904	1.170537
0.5	0.0134112242	1.498797	0.3731588	1.168780
0.5	0.0309816674	1.500794	0.3696517	1.174454
0.5	0.0715716700	1.496530	0.3701685	1.177999
0.5	0.1653398405	1.496932	0.3695208	1.186131
0.5	0.3819564761	1.520215	0.3616991	1.215537
0.5	0.8823689996	1.600224	0.3588329	1.297031
0.6	0.0004708938	1.506096	0.3687604	1.167879
0.6	0.0010878257	1.506083	0.3687255	1.168948
0.6	0.0025130186	1.506579	0.3679944	1.171756
0.6	0.0058053988	1.502293	0.3710339	1.170252
0.6	0.0134112242	1.498491	0.3732017	1.168635
0.6	0.0309816674	1.500950	0.3690645	1.175164
0.6	0.0715716700	1.496405	0.3698417	1.179209
0.6	0.1653398405	1.499511	0.3680062	1.189007
0.6	0.3819564761	1.527344	0.3612230	1.224418
0.6	0.8823689996	1.636666	0.3521408	1.330040
0.7	0.0004708938	1.506268	0.3686765	1.167828
0.7	0.0010878257	1.506561	0.3683907	1.169206
0.7	0.0025130186	1.507208	0.3675096	1.172312
0.7	0.0058053988	1.501873	0.3713604	1.170004
0.7	0.0134112242	1.498564	0.3729874	1.169241
0.7	0.0309816674	1.500817	0.3687206	1.175731
0.7	0.0715716700	1.496377	0.3695433	1.180294
0.7	0.1653398405	1.502354	0.3664678	1.192750
0.7	0.3819564761	1.534945	0.3616392	1.233717
0.7	0.8823689996	1.675549	0.3450125	1.363832
0.8	0.0004708938	1.506827	0.3683017	1.168088
0.8	0.0010878257	1.507226	0.3679172	1.169597
0.8	0.0025130186	1.508168	0.3667886	1.173097
0.8	0.0058053988	1.501305	0.3716630	1.169855
0.8	0.0134112242	1.498649	0.3726725	1.170228
0.8	0.0309816674	1.500783	0.3683693	1.176288
0.8	0.0715716700	1.496760	0.3689988	1.181463
0.8	0.1653398405	1.505700	0.3647061	1.197091
0.8	0.3819564761	1.543959	0.3614144	1.244215
0.8	0.8823689996	1.717554	0.3336224	1.398883
0.9	0.0004708938	1.507315	0.3680163	1.168197
0.9	0.0010878257	1.508009	0.3673684	1.170029

```
  0.9    0.0025130186  1.508442  0.3665187  1.173563
  0.9    0.0058053988  1.501085  0.3717485  1.169826
  0.9    0.0134112242  1.499273  0.3719913  1.171326
  0.9    0.0309816674  1.500796  0.3680230  1.176748
  0.9    0.0715716700  1.497739  0.3680851  1.182579
  0.9    0.1653398405  1.509280  0.3628649  1.201553
  0.9    0.3819564761  1.554378  0.3602316  1.255795
  0.9    0.8823689996  1.761009  0.3156752  1.432364
  1.0    0.0004708938  1.508070  0.3675070  1.168597
  1.0    0.0010878257  1.509005  0.3666790  1.170559
  1.0    0.0025130186  1.508900  0.3660798  1.174080
  1.0    0.0058053988  1.500443  0.3720585  1.169715
  1.0    0.0134112242  1.500234  0.3710372  1.172512
  1.0    0.0309816674  1.500682  0.3678118  1.177340
  1.0    0.0715716700  1.498437  0.3675010  1.183449
  1.0    0.1653398405  1.513078  0.3609402  1.206164
  1.0    0.3819564761  1.566569  0.3574504  1.267870
  1.0    0.8823689996  1.802843  0.3046895  1.461481

RMSE was used to select the optimal model using the smallest value.
The final values used for the model were alpha = 0.1 and lambda = 0.3819565.
```

```
# Coefficient of the final model. You need
# to specify the best lambda
coef(elasticTrain$finalModel, elasticTrain$bestTune$lambda)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                                      s1
(Intercept)                2.331031e+00
Duration                  -3.144057e-03
Distance                   6.956174e-05
`Max Speed`                .
`HML Distance`             2.923341e-05
`HML Efforts`              .
`Sprint Distance`         -1.594891e-05
Sprints                    .
Accelerations              1.078781e-02
Decelerations              1.312460e-02
`Average Metabolic Power`  2.306594e-01
`Dynamic Stress Load`      2.193494e-03
`High Speed Running (Relative)` .
`HML Density`              2.700968e-02
`Speed Intensity`          9.613224e-04
Impacts                    .
```

```
tLLTrainElastic <- elasticTrain$finalModel$nulldev - deviance(elasticTrain$finalModel)
deviance(elasticTrain$finalModel)
```

```
  [1] 3347.083 3300.826 3214.079 3116.592 3015.557 2922.288 2836.609 2758.342
  [9] 2687.065 2622.482 2564.226 2511.320 2463.663 2421.377 2383.422 2349.885
 [17] 2320.117 2292.082 2264.867 2240.321 2218.394 2198.717 2181.041 2163.456
 [25] 2147.073 2132.499 2119.503 2107.920 2097.627 2088.415 2080.189 2072.836
 [33] 2066.252 2060.442 2055.473 2050.333 2045.609 2041.465 2037.842 2034.562
 [41] 2030.858 2027.261 2023.933 2021.059 2017.872 2013.892 2010.382 2006.494
 [49] 2002.495 1998.785 1995.347 1992.179 1989.256 1986.575 1984.103 1981.833
 [57] 1979.733 1977.825 1976.081 1974.487 1973.033 1971.705 1970.494 1969.387
 [65] 1968.394 1967.406 1966.500 1965.818 1965.264 1964.794 1964.388 1964.027
 [73] 1963.721 1963.390 1962.844 1962.135 1961.447 1960.755 1960.040 1959.312
 [81] 1958.568 1957.800 1957.010 1956.199 1955.374 1954.521 1953.653 1952.770
```

```
 [89] 1951.875 1950.970 1950.064 1949.158 1948.250 1947.354 1946.472 1945.610
 [97] 1944.770 1943.956 1943.171 1942.418
```

```
kTrainElastic <- elasticTrain$finalModel$df
nTrainElastic <- elasticTrain$finalModel$nobs
AICcTrainElastic <- -tLLTrainElastic+2*kTrainElastic+2*kTrainElastic*(kTrainElastic+1)/(nTrainElastic-kTrainEl
AICcTrainElastic
```

```
  [1]     0.00000   -40.23148  -122.94068  -214.33864  -315.37312  -408.64264
  [7]  -494.32129  -572.58803  -643.86497  -708.44871  -766.70433  -817.57224
 [13]  -865.22947  -905.47207  -943.42690  -976.96396 -1006.73189 -1032.72004
 [19] -1059.93509 -1086.52848 -1108.45565 -1128.13279 -1145.80840 -1161.34626
 [25] -1179.77645 -1194.35076 -1207.34671 -1218.92923 -1229.22223 -1238.43472
 [31] -1246.66015 -1254.01355 -1260.59776 -1268.44958 -1273.41920 -1276.51590
 [37] -1281.24034 -1285.38397 -1289.00764 -1290.24077 -1293.94483 -1297.54090
 [43] -1300.86900 -1303.74296 -1302.82358 -1306.80367 -1310.31358 -1312.14076
 [49] -1316.13989 -1319.84964 -1323.28832 -1326.45598 -1329.37886 -1332.05994
 [55] -1334.53237 -1336.80261 -1338.90248 -1340.80981 -1342.55412 -1344.14766
 [61] -1345.60236 -1346.92988 -1348.14147 -1349.24797 -1350.24136 -1351.22910
 [67] -1352.13560 -1354.87772 -1355.43170 -1355.90094 -1356.30703 -1356.66780
 [73] -1354.91392 -1353.18024 -1353.72693 -1354.43576 -1355.12320 -1355.81560
 [79] -1356.53016 -1357.25833 -1358.00258 -1358.77057 -1359.56056 -1360.37154
 [85] -1361.19694 -1362.04916 -1362.91754 -1363.80051 -1364.69581 -1365.60063
 [91] -1366.50683 -1367.41296 -1368.32092 -1369.21639 -1370.09833 -1370.96026
 [97] -1371.80099 -1372.61454 -1373.40002 -1374.15230
```

```
get_best_result(elasticTrain)
```

```
  alpha    lambda     RMSE  Rsquared      MAE     RMSESD RsquaredSD      MAESD
1   0.1 0.3819565 1.491124 0.3738442 1.183919 0.09112329 0.05791713 0.07275976
```

```
# # Make predictions on the test data
# x.test <- model.matrix(RPE ~., test.data)[,-1]
# predictions <- model %>% predict(x.test)
# # Model performance metrics
# data.frame(
#   RMSE = RMSE(predictions, test.data$RPE),
#   Rsquare = R2(predictions, test.data$RPE)
# )
```

```
########################################Using Ridge, Lasso, and Elastic Net in One Computation
```

```
#Computing all 3
lambda <- 10^seq(-3, 3, length = 100)
```

```
# Build the model
set.seed(42)
# ridge <- train(
#   RPE ~., data = train.data, method = "glmnet",
#   trControl = trainControl("cv", number = 10),
#   tuneGrid = expand.grid(alpha = 0, lambda = lambda)
# )
# Model coefficients
# coef(ridge$finalModel, ridge$bestTune$lambda)
# # Make predictions
# predictions <- ridge %>% predict(test.data)
# # Model prediction performance
# data.frame(
```

```
#   RMSE = RMSE(predictions, test.data$RPE),
#   Rsquare = R2(predictions, test.data$RPE)
# )

# Build the model
set.seed(42)
# lasso <- train(
#   RPE ~., data = train.data, method = "glmnet",
#   trControl = trainControl("cv", number = 10),
#   tuneGrid = expand.grid(alpha = 1, lambda = lambda)
# )
# Model coefficients
# coef(lasso$finalModel, lasso$bestTune$lambda)
# # Make predictions
# predictions <- lasso %>% predict(test.data)
# # Model prediction performance
# data.frame(
#   RMSE = RMSE(predictions, test.data$RPE),
#   Rsquare = R2(predictions, test.data$RPE)
# )

# Build the model
set.seed(42)
# elastic <- train(
#   RPE ~., data = train.data, method = "glmnet",
#   trControl = trainControl("cv", number = 10),
#   tuneLength = 10
# )
# Model coefficients
# coef(elastic$finalModel, elastic$bestTune$lambda)
# # Make predictions
# predictions <- elastic %>% predict(test.data)
# # Model prediction performance
# data.frame(
#   RMSE = RMSE(predictions, test.data$RPE),
#   Rsquare = R2(predictions, test.data$RPE)
# )

# #Comparing models
# models <- list(ridge = ridge, lasso = lasso, elastic = elastic)
# resamples(models) %>% summary( metric = "RMSE")

#Lasso is best model since it has the lowest median RMSE

########################################SVM
start_time <- Sys.time()
svm.model <- svm(RPE ~ ., data = res$train, cost = 100, gamma = 1)
end_time <- Sys.time()
time <- end_time - start_time
time

Time difference of 0.06777215 secs

svm.model

Call:
svm(formula = RPE ~ ., data = res$train, cost = 100, gamma = 1)
```

```
Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  100
      gamma:  1
    epsilon:  0.1


Number of Support Vectors:  852
```

```
plot(svm.model, res$train)
```

```
svmTrain <- fit(RPE ~ ., data = res$train, model = "svm", kpar=list(sigma=0.10), C=2)
svmImp <- Importance(svmTrain, data = res$train)
svmImp$interactions
```

```
NULL
```

```
print(round(svmImp$imp,digits = 2))
```

```
 [1] 0.00 0.10 0.10 0.05 0.07 0.09 0.03 0.09 0.05 0.06 0.09 0.08 0.03 0.07 0.05
[16] 0.03
```

```
# tune.svm(svm.model)
# tune(svm.model)
W <- t(svm.model$coefs) %*% svm.model$SV
sort(W)
```

```
 [1]   9.137915  23.207070  40.650941  43.920135  68.034881 121.701904
 [7] 137.838199 142.055588 163.033720 167.400299 169.518832 175.884899
[13] 191.000272 192.541981 193.958701
```

```
svm.model
```

```
Call:
svm(formula = RPE ~ ., data = res$train, cost = 100, gamma = 1)


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  100
      gamma:  1
    epsilon:  0.1


Number of Support Vectors:  852
```

```
b <- svm.model$rho
b
```

```
[1] -0.3231657
```

```
OptModelsvm <- tune(svm, RPE~., data=res$train,ranges=list(elsilon=seq(0,1,0.1), cost=1:100))
OptModelsvm
```

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 elsilon cost
       0    1

- best performance: 1.797541
```

```
bestSVM <- OptModelsvm$best.model
bestSVM
```

```

Call:
best.tune(method = svm, train.x = RPE ~ ., data = res$train, ranges = list(elsilon = seq(0,
    1, 0.1), cost = 1:100))


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.06666667
    epsilon:  0.1


Number of Support Vectors:  839
```

```
# Rsq.ad(svm.model)
2*RMSE(svm.model)^2 / RMSE(svm.model)
```

```
[1] 0.3625336
```

```
RMSE(bestSVM)
```

```
[1] 1.176771
```

```
aicSVM <- ((2*10) - 16*log(RMSE(svm.model)))*16
aicSVM
```

```
[1] 757.193
```

```
training <- res$train
training[["RPE"]] <- factor(training[["RPE"]])
summary(training)
```

```
      RPE         Duration         Distance          Max Speed
 4      :205   Min.   : 20.00   Min.   :   19.45   Min.   : 4.68
 5      :171   1st Qu.: 80.00   1st Qu.: 3499.15   1st Qu.:21.17
 6      :163   Median : 93.00   Median : 4884.67   Median :23.44
 7      :124   Mean   : 98.46   Mean   : 5153.64   Mean   :23.42
 3      : 96   3rd Qu.:111.00   3rd Qu.: 6223.97   3rd Qu.:25.31
 2      : 63   Max.   :243.00   Max.   :15642.70   Max.   :43.20
 (Other):131
  HML Distance     HML Efforts     Sprint Distance       Sprints
 Min.   :   0.0   Min.   :  0.00   Min.   :    0.00   Min.   : 0.000
 1st Qu.: 416.8   1st Qu.: 51.00   1st Qu.:    8.23   1st Qu.: 1.000
```

```
 Median : 726.5   Median : 88.00   Median :   53.14   Median : 4.000
 Mean   : 849.3   Mean   : 99.72   Mean   :  164.60   Mean   : 6.033
 3rd Qu.:1141.1   3rd Qu.:138.00   3rd Qu.:  158.71   3rd Qu.: 9.000
 Max.   :6579.3   Max.   :451.00   Max.   :14355.40   Max.   :41.000

 Accelerations    Decelerations    Average Metabolic Power Dynamic Stress Load
 Min.   :  0.00   Min.   :  0.00   Min.   :0.010           Min.   :  0.82
 1st Qu.: 21.00   1st Qu.: 18.00   1st Qu.:2.870           1st Qu.: 77.36
 Median : 33.00   Median : 30.00   Median :3.670           Median :128.60
 Mean   : 34.68   Mean   : 32.25   Mean   :3.816           Mean   :152.98
 3rd Qu.: 47.00   3rd Qu.: 44.00   3rd Qu.:4.630           3rd Qu.:203.39
 Max.   :110.00   Max.   :105.00   Max.   :7.340           Max.   :760.88

 High Speed Running (Relative)  HML Density      Speed Intensity
 Min.   :   0.00               Min.   :  0.000   Min.   :  0.79
 1st Qu.:   4.31               1st Qu.:  6.980   1st Qu.:160.58
 Median :  26.60               Median :  7.780   Median :225.23
 Mean   :  93.47               Mean   :  9.031   Mean   :243.13
 3rd Qu.:  86.64               3rd Qu.:  8.870   3rd Qu.:295.56
 Max.   :5347.05               Max.   :100.320   Max.   :838.88

    Impacts
 Min.   :  0.00
 1st Qu.: 12.00
 Median : 28.00
 Mean   : 41.98
 3rd Qu.: 62.00
 Max.   :170.00
```

```
set.seed(42)
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

svm_Linear <- train(RPE ~., data = res$train, method = "svmLinear",
                    trControl=trctrl,
                    preProcess = c("center", "scale"),
                    tuneLength = 10)
svm_Linear
```

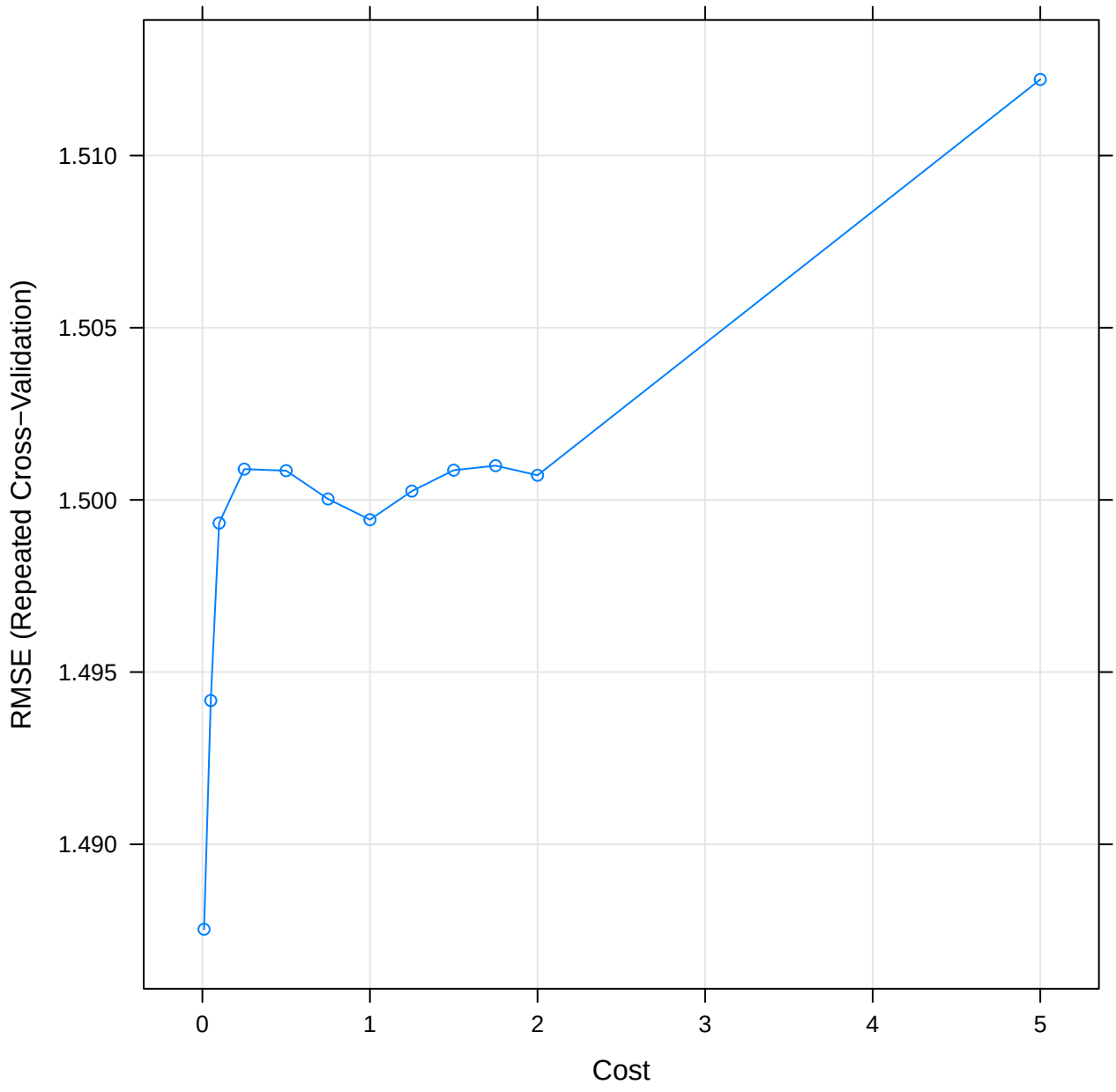
```
Support Vector Machines with Linear Kernel

953 samples
 15 predictor

Pre-processing: centered (15), scaled (15)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 858, 857, 858, 859, 858, 857, ...
Resampling results:

  RMSE      Rsquared   MAE
  1.498065  0.3788261  1.163678

Tuning parameter 'C' was held constant at a value of 1
```

```
test_predSVM <- predict(svm_Linear, newdata = res$test)

u <- union(test_predSVM, res$test$RPE)
t <- table(factor(test_predSVM, u), factor(res$test$RPE, u))
#confusionMatrix(t)
```

```
start_time <- Sys.time()
grid <- expand.grid(C = c(0,0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2,5))
svm_Linear_Grid <- train(RPE ~., data = res$train, method = "svmLinear",
                         trControl=trctrl,
                         preProcess = c("center", "scale"),
                         tuneGrid = grid,
                         tuneLength = 10)
svm_Linear_Grid #0.01 is best
```

```
Support Vector Machines with Linear Kernel

953 samples
 15 predictor

Pre-processing: centered (15), scaled (15)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 858, 858, 858, 858, 856, 858, ...
Resampling results across tuning parameters:

  C     RMSE      Rsquared   MAE
  0.00       NaN        NaN       NaN
  0.01  1.487530  0.3870936  1.161808
  0.05  1.494173  0.3896386  1.157991
  0.10  1.499326  0.3874287  1.159787
  0.25  1.500894  0.3871135  1.160358
  0.50  1.500848  0.3870583  1.160079
  0.75  1.500026  0.3878022  1.159029
  1.00  1.499425  0.3886124  1.158399
  1.25  1.500255  0.3887235  1.157691
  1.50  1.500863  0.3890451  1.157434
  1.75  1.500994  0.3891002  1.157326
  2.00  1.500715  0.3891714  1.157018
  5.00  1.512209  0.3834293  1.159519

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was C = 0.01.
```

```
plot(svm_Linear_Grid)
```




```
end_time <- Sys.time()
time <- end_time - start_time
time
```

```
Time difference of 55.4624 mins
```

```
test_pred_grid <- predict(svm_Linear_Grid, newdata = res$test)
head(test_pred_grid)
```

```
       4        6        7        8       11       30
5.179716 5.749075 6.364476 6.489117 5.284881 4.510583
```

```
u1 <- union(test_pred_grid, res$test$RPE)
t1 <- table(factor(test_pred_grid, u1), factor(res$test$RPE, u1))
#confusionMatrix(t1)
```

```
########################################Boosted Forests
```

```
#Fitting the model on training set
```

```r
set.seed(42)
boostedTrain <- train(
  RPE ~., data = res$train, method = "xgbTree",
  trControl = trainControl("cv", number = 10), verbosity = 0
)
# Best tuning parameter
boostedTrain$bestTune
```

```
   nrounds max_depth eta gamma colsample_bytree min_child_weight subsample
34      50         2 0.3     0              0.8                1         1
```

```r
summary(boostedTrain$results)
```

```
      eta          max_depth     gamma   colsample_bytree min_child_weight
 Min.   :0.30   Min.   :1   Min.   :0   Min.   :0.6      Min.   :1
 1st Qu.:0.30   1st Qu.:1   1st Qu.:0   1st Qu.:0.6      1st Qu.:1
 Median :0.35   Median :2   Median :0   Median :0.7      Median :1
 Mean   :0.35   Mean   :2   Mean   :0   Mean   :0.7      Mean   :1
 3rd Qu.:0.40   3rd Qu.:3   3rd Qu.:0   3rd Qu.:0.8      3rd Qu.:1
 Max.   :0.40   Max.   :3   Max.   :0   Max.   :0.8      Max.   :1
   subsample       nrounds         RMSE          Rsquared           MAE
 Min.   :0.50   Min.   : 50   Min.   :1.330   Min.   :0.4084   Min.   :1.034
 1st Qu.:0.50   1st Qu.: 50   1st Qu.:1.364   1st Qu.:0.4565   1st Qu.:1.071
 Median :0.75   Median :100   Median :1.381   Median :0.4680   Median :1.086
 Mean   :0.75   Mean   :100   Mean   :1.390   Mean   :0.4642   Mean   :1.090
 3rd Qu.:1.00   3rd Qu.:150   3rd Qu.:1.403   3rd Qu.:0.4775   3rd Qu.:1.100
 Max.   :1.00   Max.   :150   Max.   :1.515   Max.   :0.5025   Max.   :1.196
     RMSESD          RsquaredSD          MAESD
 Min.   :0.06266   Min.   :0.05519   Min.   :0.03205
 1st Qu.:0.08670   1st Qu.:0.07758   1st Qu.:0.05178
 Median :0.09772   Median :0.08161   Median :0.06007
 Mean   :0.09608   Mean   :0.08136   Mean   :0.05944
 3rd Qu.:0.10474   3rd Qu.:0.08758   3rd Qu.:0.06581
 Max.   :0.14765   Max.   :0.09885   Max.   :0.09874
```

```r
boostedTrain$results
```

```
    eta max_depth gamma colsample_bytree min_child_weight subsample nrounds
1   0.3         1     0              0.6                1      0.50      50
4   0.3         1     0              0.6                1      0.75      50
7   0.3         1     0              0.6                1      1.00      50
10  0.3         1     0              0.8                1      0.50      50
13  0.3         1     0              0.8                1      0.75      50
16  0.3         1     0              0.8                1      1.00      50
55  0.4         1     0              0.6                1      0.50      50
58  0.4         1     0              0.6                1      0.75      50
61  0.4         1     0              0.6                1      1.00      50
64  0.4         1     0              0.8                1      0.50      50
67  0.4         1     0              0.8                1      0.75      50
70  0.4         1     0              0.8                1      1.00      50
19  0.3         2     0              0.6                1      0.50      50
22  0.3         2     0              0.6                1      0.75      50
25  0.3         2     0              0.6                1      1.00      50
28  0.3         2     0              0.8                1      0.50      50
31  0.3         2     0              0.8                1      0.75      50
34  0.3         2     0              0.8                1      1.00      50
73  0.4         2     0              0.6                1      0.50      50
76  0.4         2     0              0.6                1      0.75      50
79  0.4         2     0              0.6                1      1.00      50
```

82	0.4	2	0	0.8	1	0.50	50
85	0.4	2	0	0.8	1	0.75	50
88	0.4	2	0	0.8	1	1.00	50
37	0.3	3	0	0.6	1	0.50	50
40	0.3	3	0	0.6	1	0.75	50
43	0.3	3	0	0.6	1	1.00	50
46	0.3	3	0	0.8	1	0.50	50
49	0.3	3	0	0.8	1	0.75	50
52	0.3	3	0	0.8	1	1.00	50
91	0.4	3	0	0.6	1	0.50	50
94	0.4	3	0	0.6	1	0.75	50
97	0.4	3	0	0.6	1	1.00	50
100	0.4	3	0	0.8	1	0.50	50
103	0.4	3	0	0.8	1	0.75	50
106	0.4	3	0	0.8	1	1.00	50
2	0.3	1	0	0.6	1	0.50	100
5	0.3	1	0	0.6	1	0.75	100
8	0.3	1	0	0.6	1	1.00	100
11	0.3	1	0	0.8	1	0.50	100
14	0.3	1	0	0.8	1	0.75	100
17	0.3	1	0	0.8	1	1.00	100
56	0.4	1	0	0.6	1	0.50	100
59	0.4	1	0	0.6	1	0.75	100
62	0.4	1	0	0.6	1	1.00	100
65	0.4	1	0	0.8	1	0.50	100
68	0.4	1	0	0.8	1	0.75	100
71	0.4	1	0	0.8	1	1.00	100
20	0.3	2	0	0.6	1	0.50	100
23	0.3	2	0	0.6	1	0.75	100
26	0.3	2	0	0.6	1	1.00	100
29	0.3	2	0	0.8	1	0.50	100
32	0.3	2	0	0.8	1	0.75	100
35	0.3	2	0	0.8	1	1.00	100
74	0.4	2	0	0.6	1	0.50	100
77	0.4	2	0	0.6	1	0.75	100
80	0.4	2	0	0.6	1	1.00	100
83	0.4	2	0	0.8	1	0.50	100
86	0.4	2	0	0.8	1	0.75	100
89	0.4	2	0	0.8	1	1.00	100
38	0.3	3	0	0.6	1	0.50	100
41	0.3	3	0	0.6	1	0.75	100
44	0.3	3	0	0.6	1	1.00	100
47	0.3	3	0	0.8	1	0.50	100
50	0.3	3	0	0.8	1	0.75	100
53	0.3	3	0	0.8	1	1.00	100
92	0.4	3	0	0.6	1	0.50	100
95	0.4	3	0	0.6	1	0.75	100
98	0.4	3	0	0.6	1	1.00	100
101	0.4	3	0	0.8	1	0.50	100
104	0.4	3	0	0.8	1	0.75	100
107	0.4	3	0	0.8	1	1.00	100
3	0.3	1	0	0.6	1	0.50	150
6	0.3	1	0	0.6	1	0.75	150
9	0.3	1	0	0.6	1	1.00	150
12	0.3	1	0	0.8	1	0.50	150
15	0.3	1	0	0.8	1	0.75	150
18	0.3	1	0	0.8	1	1.00	150
57	0.4	1	0	0.6	1	0.50	150
60	0.4	1	0	0.6	1	0.75	150
63	0.4	1	0	0.6	1	1.00	150

66	0.4	1	0	0.8	1	0.50	150	
69	0.4	1	0	0.8	1	0.75	150	
72	0.4	1	0	0.8	1	1.00	150	
21	0.3	2	0	0.6	1	0.50	150	
24	0.3	2	0	0.6	1	0.75	150	
27	0.3	2	0	0.6	1	1.00	150	
30	0.3	2	0	0.8	1	0.50	150	
33	0.3	2	0	0.8	1	0.75	150	
36	0.3	2	0	0.8	1	1.00	150	
75	0.4	2	0	0.6	1	0.50	150	
78	0.4	2	0	0.6	1	0.75	150	
81	0.4	2	0	0.6	1	1.00	150	
84	0.4	2	0	0.8	1	0.50	150	
87	0.4	2	0	0.8	1	0.75	150	
90	0.4	2	0	0.8	1	1.00	150	
39	0.3	3	0	0.6	1	0.50	150	
42	0.3	3	0	0.6	1	0.75	150	
45	0.3	3	0	0.6	1	1.00	150	
48	0.3	3	0	0.8	1	0.50	150	
51	0.3	3	0	0.8	1	0.75	150	
54	0.3	3	0	0.8	1	1.00	150	
93	0.4	3	0	0.6	1	0.50	150	
96	0.4	3	0	0.6	1	0.75	150	
99	0.4	3	0	0.6	1	1.00	150	
102	0.4	3	0	0.8	1	0.50	150	
105	0.4	3	0	0.8	1	0.75	150	
108	0.4	3	0	0.8	1	1.00	150	

	RMSE	Rsquared	MAE	RMSESD	RsquaredSD	MAESD
1	1.377501	0.4630157	1.095023	0.09656523	0.08823335	0.06175972
4	1.381158	0.4611159	1.100182	0.07160974	0.07380359	0.03986108
7	1.383346	0.4587973	1.098311	0.09225706	0.08416694	0.05276516
10	1.396781	0.4474767	1.104751	0.10530667	0.09572929	0.06400879
13	1.366587	0.4724312	1.084674	0.08887270	0.07519117	0.04996330
16	1.382724	0.4601403	1.100052	0.09726796	0.08654707	0.06387694
55	1.386618	0.4558717	1.099905	0.09995073	0.06753842	0.06344086
58	1.368071	0.4712240	1.085137	0.09021239	0.08792024	0.05477206
61	1.382160	0.4606727	1.095918	0.10313586	0.08813390	0.06404095
64	1.399467	0.4498492	1.100863	0.10461942	0.07372302	0.06061579
67	1.383194	0.4573059	1.091495	0.11184971	0.09291200	0.06645550
70	1.380660	0.4618523	1.093531	0.10386883	0.08977275	0.07048333
19	1.356570	0.4818284	1.070663	0.09873887	0.07772045	0.05207916
22	1.351870	0.4849047	1.054607	0.10511659	0.09769821	0.05735971
25	1.346354	0.4891837	1.050412	0.10219348	0.08192495	0.04917067
28	1.361687	0.4793381	1.072504	0.08390615	0.07926850	0.05649664
31	1.337134	0.4943403	1.043052	0.07790242	0.07866347	0.03205018
34	1.329680	0.5004681	1.037810	0.09572625	0.08075217	0.05049435
73	1.395526	0.4586004	1.094474	0.06983182	0.06974709	0.03836719
76	1.349787	0.4869329	1.058717	0.09926810	0.08706828	0.05847498
79	1.352975	0.4852243	1.054426	0.10781648	0.08724988	0.06067293
82	1.431956	0.4375249	1.132625	0.10725986	0.08756571	0.06310154
85	1.359423	0.4826355	1.063642	0.10166454	0.08144927	0.04060012
88	1.366440	0.4758089	1.062973	0.10164763	0.08497984	0.06562271
37	1.389678	0.4628284	1.093854	0.08428090	0.08156752	0.05936431
40	1.353390	0.4867349	1.054604	0.06266236	0.07972238	0.03796107
43	1.343076	0.4945955	1.048590	0.07688209	0.06195036	0.04931702
46	1.441273	0.4360861	1.124491	0.09989002	0.07113586	0.05796168
49	1.382843	0.4684020	1.076692	0.09795437	0.08053932	0.06649756
52	1.341484	0.4940653	1.039911	0.09749003	0.08445553	0.06210244
91	1.414754	0.4592713	1.118037	0.12989631	0.08992224	0.07773443
94	1.418418	0.4456534	1.101916	0.07179849	0.07264181	0.05585668

97	1.367199	0.4800313	1.084851	0.10535240	0.08484346	0.06011990
100	1.426593	0.4505973	1.119891	0.08944627	0.07546406	0.06190329
103	1.402366	0.4553178	1.090969	0.08568054	0.07766986	0.06501765
106	1.356199	0.4852915	1.058872	0.09348760	0.08104800	0.05942529
2	1.370074	0.4736604	1.084366	0.10754955	0.09052418	0.05686188
5	1.364637	0.4750867	1.078403	0.07762767	0.07777149	0.04182446
8	1.360601	0.4762136	1.073928	0.09072626	0.08673698	0.04474539
11	1.378665	0.4645965	1.083361	0.11091939	0.08762624	0.05845529
14	1.356692	0.4810912	1.068463	0.09852940	0.08167874	0.04994782
17	1.364401	0.4736315	1.078695	0.09556042	0.08853821	0.05677944
56	1.375855	0.4675700	1.086412	0.08634780	0.05946495	0.06494486
59	1.350482	0.4859031	1.065085	0.08184156	0.08142638	0.04348452
62	1.367957	0.4718070	1.079326	0.09862681	0.08638055	0.05514596
65	1.392095	0.4581953	1.094365	0.08878908	0.07065809	0.04873406
68	1.388886	0.4585186	1.092188	0.10981626	0.08988964	0.05429531
71	1.371573	0.4687360	1.079732	0.09988901	0.08845560	0.06020783
20	1.391893	0.4604970	1.100487	0.12099582	0.09520272	0.07830824
23	1.349597	0.4915267	1.049781	0.09964660	0.08366937	0.06122380
26	1.361750	0.4820276	1.057662	0.09446699	0.08037474	0.05819825
29	1.393946	0.4627677	1.096548	0.09973303	0.08533971	0.05910194
32	1.355408	0.4846031	1.063051	0.08538895	0.07961144	0.04667121
35	1.330140	0.5025335	1.034314	0.08131830	0.07813656	0.05360044
74	1.415986	0.4511820	1.108641	0.06533519	0.06172874	0.03787386
77	1.359256	0.4828028	1.062154	0.11679228	0.08607984	0.06402867
80	1.376329	0.4727733	1.065306	0.10907954	0.09138552	0.07286640
83	1.468540	0.4185851	1.155789	0.10811679	0.09420050	0.06864901
86	1.376690	0.4749820	1.078277	0.09314402	0.07729256	0.04541956
89	1.385508	0.4657835	1.078610	0.10397424	0.07773345	0.07594156
38	1.413934	0.4539172	1.106531	0.08098604	0.08612638	0.04760102
41	1.381544	0.4728140	1.080904	0.06321398	0.07713852	0.04400278
44	1.362599	0.4827886	1.061336	0.08681210	0.07070735	0.06537462
47	1.462059	0.4275870	1.145616	0.10315126	0.06348198	0.07706388
50	1.415819	0.4508215	1.095705	0.11424834	0.09225760	0.07686005
53	1.369021	0.4781325	1.060992	0.10188723	0.08374194	0.07098157
92	1.472297	0.4328591	1.161578	0.12376436	0.09511003	0.08565618
95	1.460026	0.4260843	1.131959	0.08127014	0.07103073	0.06167870
98	1.381735	0.4738625	1.092963	0.11678086	0.09038162	0.07831766
101	1.488789	0.4193890	1.168807	0.07995891	0.07092580	0.06031471
104	1.462431	0.4229731	1.134154	0.08972219	0.07881545	0.06192667
107	1.382064	0.4732476	1.082503	0.10444108	0.08047035	0.06636556
3	1.374360	0.4718839	1.084065	0.10090262	0.08164062	0.05587675
6	1.366414	0.4758478	1.076740	0.07501361	0.07958500	0.03425577
9	1.361815	0.4756044	1.070046	0.08448504	0.08541299	0.03887895
12	1.393789	0.4570491	1.090574	0.11499630	0.09053024	0.06078499
15	1.355064	0.4842969	1.068288	0.10121685	0.08057330	0.05088659
18	1.359518	0.4774622	1.070652	0.09418915	0.08868359	0.04855388
57	1.374321	0.4707621	1.086052	0.09848463	0.06832372	0.06862565
60	1.371198	0.4722623	1.073997	0.08973976	0.08619730	0.04958088
63	1.365439	0.4751879	1.077207	0.09692536	0.08314609	0.05439946
66	1.391866	0.4576079	1.097091	0.08974391	0.07088962	0.05299244
69	1.386227	0.4616049	1.087865	0.11333777	0.08844856	0.05832306
72	1.368218	0.4716376	1.075440	0.09739539	0.08548342	0.05339283
21	1.406483	0.4548437	1.112059	0.10597620	0.08447025	0.07229013
24	1.364368	0.4844181	1.067732	0.10006751	0.07906086	0.06251748
27	1.375585	0.4756050	1.073384	0.09206853	0.08158475	0.06511885
30	1.395590	0.4651542	1.094800	0.10176026	0.08909960	0.07235897
33	1.378548	0.4729856	1.078015	0.08276907	0.06965124	0.04767365
36	1.348892	0.4908540	1.051447	0.08994611	0.08345473	0.06870646
75	1.447033	0.4357076	1.131563	0.06515616	0.05854784	0.04795893
78	1.372623	0.4775387	1.075775	0.10903207	0.08039149	0.06644501

```
81  1.404044 0.4573952 1.085939 0.10201453 0.08695547 0.06337397
84  1.463528 0.4244659 1.146662 0.11166400 0.09586164 0.06834695
87  1.401071 0.4618750 1.089245 0.09674479 0.07354057 0.05650340
90  1.403275 0.4566755 1.090439 0.11140712 0.07965466 0.08405930
39  1.436524 0.4413146 1.129359 0.07794057 0.08784339 0.04438619
42  1.394874 0.4648662 1.086453 0.07721007 0.08139370 0.05282797
45  1.374243 0.4763450 1.073059 0.08601190 0.06893765 0.06001780
48  1.479944 0.4196120 1.156240 0.09208812 0.05518966 0.07323765
51  1.434420 0.4426849 1.112545 0.10707166 0.08713974 0.07358473
54  1.384061 0.4697625 1.073676 0.10454364 0.08324989 0.07430438
93  1.497492 0.4239323 1.185792 0.14764774 0.09885071 0.09873583
96  1.467758 0.4249403 1.136761 0.07251784 0.06329599 0.05531554
99  1.409749 0.4588830 1.114347 0.12237565 0.09407634 0.08627505
102 1.515319 0.4083773 1.196222 0.08929479 0.07933863 0.06944521
105 1.475780 0.4166515 1.151643 0.08705779 0.08101087 0.06427348
108 1.403067 0.4619954 1.100201 0.10626379 0.07708676 0.06247888
```

```
boostedTrain$bestTune
```

```
   nrounds max_depth eta gamma colsample_bytree min_child_weight subsample
34      50         2 0.3     0              0.8                1         1
```

```
boostedTrain$finalModel
```

```
##### xgb.Booster
raw: 38.8 Kb
call:
  xgboost::xgb.train(params = list(eta = param$eta, max_depth = param$max_depth,
    gamma = param$gamma, colsample_bytree = param$colsample_bytree,
    min_child_weight = param$min_child_weight, subsample = param$subsample),
    data = x, nrounds = param$nrounds, objective = "reg:squarederror",
    verbosity = 0)
params (as set within xgb.train):
  eta = "0.3", max_depth = "2", gamma = "0", colsample_bytree = "0.8", min_child_weight = "1", subsample = "1"
xgb.attributes:
  niter
callbacks:
  cb.print.evaluation(period = print_every_n)
# of features: 15
niter: 50
nfeatures : 15
xNames : Duration Distance ‘Max Speed‘ ‘HML Distance‘ ‘HML Efforts‘ ‘Sprint Distance‘ Sprints Accelerations De
problemType : Regression
tuneValue :
       nrounds max_depth eta gamma colsample_bytree min_child_weight subsample
34      50         2 0.3     0              0.8                1         1
obsLevels : NA
param :
    $verbosity
[1] 0
```

```
aicBoosted <- ((2*8) - 16*log(RMSE(boostedTrain)))*16
aicBoosted
```

```
[1] 239.3925
```

```
#Variable importance
#varImp(boostedTrain$bestTune)
varImp(boostedTrain)
```

```
xgbTree variable importance

                               Overall
`HML Distance`                100.0000
`Average Metabolic Power`      75.7709
Accelerations                  57.5361
Duration                       24.4045
`Speed Intensity`              23.4925
Decelerations                  20.0878
`Sprint Distance`              19.9347
`Dynamic Stress Load`          19.3045
`HML Density`                  13.9003
`Max Speed`                     8.0946
`HML Efforts`                   7.3055
`High Speed Running (Relative)` 5.5182
Distance                        2.3922
Sprints                         0.7249
Impacts                         0.0000
```

```
#varImp(boostedTrain$finalModel)
xgb.importance(names(res$train[,2:16]), boostedTrain$finalModel)
```

```
                        Feature        Gain      Cover  Frequency
 1:                 HML Distance 0.255904006 0.06680309 0.07482993
 2:       Average Metabolic Power 0.194581149 0.12450811 0.13605442
 3:                 Accelerations 0.148429360 0.08285849 0.08163265
 4:                     Duration 0.064574528 0.12297602 0.11564626
 5:               Speed Intensity 0.062266389 0.04286689 0.04081633
 6:                Decelerations 0.053649212 0.07530301 0.07482993
 7:               Sprint Distance 0.053261580 0.06636235 0.06802721
 8:           Dynamic Stress Load 0.051666544 0.09934414 0.07482993
 9:                   HML Density 0.037988707 0.08729734 0.06802721
10:                     Max Speed 0.023294753 0.06308830 0.06802721
11:                   HML Efforts 0.021297696 0.03487066 0.04761905
12: High Speed Running (Relative) 0.016774000 0.08191406 0.08163265
13:                      Distance 0.008862215 0.01280235 0.02721088
14:                       Sprints 0.004642215 0.02127079 0.02721088
15:                       Impacts 0.002807645 0.01773440 0.01360544
```

```
?xgb.importance
#shapeley score/value
```

```
RMSE(boostedTrain)
```

```
[1] 1.067023
```

```
# Make predictions on the test data
predictions <- boostedTrain %>% predict(res$test)
head(predictions)
```

```
[1] 5.815962 6.560360 7.339036 7.218025 6.125978 5.827328
```

```
# Compute model prediction accuracy rate
cor(predictions, res$test$RPE)
```

```
[1] 0.7473546
```

```
# # Compute the average prediction error RMSE
# RMSE(predictions, test.data$RPE)
#compare
#RMSE(predictions, res$train$RPE) #test is not higher than train, no evidence of overfit
?cor
```

```
Help on topic 'cor' was found in the following packages:

  Package               Library
  mosaic                /Library/Frameworks/R.framework/Versions/4.1-arm64/Resources/library
  stats                 /Library/Frameworks/R.framework/Versions/4.1-arm64/Resources/library


Using the first match ...
```

```
######################################## KNN
#scale

train.loan <- res$train # 70% training data
test.loan <- res$test # remaining 30% test data

#Creating seperate dataframe for rpe feature which is the target.
train.loan_labels <- res$train[,1]
head(train.loan_labels)
```

```
[1]  4 10  6  7  7  8
```

```
test.loan_labels <-res$test[,1]
head(test.loan_labels)
```

```
[1] 8 7 6 7 5 8
```

```
#Find the number of observation
NROW(train.loan_labels) #30.85, rows 952
```

```
[1] 953
```

```
?knn
```

```
Help on topic 'knn' was found in the following packages:

  Package               Library
  igraph                /Library/Frameworks/R.framework/Versions/4.1-arm64/Resources/library
  class                 /Library/Frameworks/R.framework/Versions/4.1-arm64/Resources/library


Using the first match ...
```

```
knn.30 <- knn(train=train.loan, test=test.loan, cl=train.loan_labels, k=30)
knn.31 <- knn(train=train.loan, test=test.loan, cl=train.loan_labels, k=31)

#Calculate the proportion of correct classification for k = 30, 31
ACC.30 <- 100 * sum(test.loan_labels == knn.30)/NROW(test.loan_labels)
ACC.31 <- 100 * sum(test.loan_labels == knn.31)/NROW(test.loan_labels)
```

```
ACC.30
```

```
[1] 29.87421
```

```
ACC.31
```

```
[1] 29.24528
```

```
# Check prediction against actual value in tabular form for k=30
table(knn.30 ,test.loan_labels)
```

```
      test.loan_labels
knn.30  0  1  2  3 3.5  4 4.5  5  6 6.5  7  8 8.5  9
   0    0  0  0  0   0  0   0  0  0   0  0  0   0  0
   1    0  1  3  1   0  1   0  1  0   0  0  0   0  0
   2    0  2  7  4   0  6   0  1  0   0  1  1   0  0
   2.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   3    2  1  6  7   1  5   0  4  2   0  0  0   0  1
   3.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   4    0  3  2 16   0 42   1 39 12   0 14  5   0  0
   4.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   5    0  1  0  1   0  9   0 12 10   1 16  4   0  2
   5.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   6    0  0  1  0   0  4   0 13  7   0  4  2   0  2
   6.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   7    0  0  0  1   0  2   0  4  4   0 15  3   2  1
   7.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   8    0  0  0  0   0  0   0  0  0   0  1  4   0  0
   8.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   9    0  0  0  0   0  0   0  0  0   0  0  0   0  0
   10   0  0  0  0   0  0   0  0  0   0  0  0   0  0
```

```
cor(as.numeric(knn.30), test.loan_labels)
```

```
[1] 0.5546072
```

```
# Check prediction against actual value in tabular form for k=31
table(knn.31 ,test.loan_labels)
```

```
      test.loan_labels
knn.31  0  1  2  3 3.5  4 4.5  5  6 6.5  7  8 8.5  9
   0    0  0  0  0   0  0   0  0  0   0  0  0   0  0
   1    0  1  2  2   0  0   0  1  0   0  0  0   0  0
   2    0  2  8  3   0  5   0  1  0   0  0  1   0  0
   2.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   3    2  1  6  6   1  6   0  3  2   0  0  0   0  1
   3.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   4    0  4  3 17   0 42   1 39 12   0 16  5   0  0
   4.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   5    0  0  0  1   0 10   0 13 10   1 12  4   0  2
   5.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   6    0  0  0  0   0  5   0 12  7   0  7  1   0  2
   6.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   7    0  0  0  1   0  1   0  5  4   0 15  7   2  1
   7.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   8    0  0  0  0   0  0   0  0  0   0  1  1   0  0
   8.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
   9    0  0  0  0   0  0   0  0  0   0  0  0   0  0
   10   0  0  0  0   0  0   0  0  0   0  0  0   0  0
```

```
cor(as.numeric(knn.31), test.loan_labels)
```

```
[1] 0.576481
```

```
#confusionMatrix(table(knn.31 ,test.loan_labels))
uKNN <- union(knn.31, res$test$RPE)
tKNN <- table(factor(knn.31, uKNN), factor(res$test$RPE, uKNN))
confusionMatrix(tKNN)
```

```
Confusion Matrix and Statistics


      4  7  5  6  3  1  2  8  9 6.5  0 8.5 4.5 3.5
  4  42 16 39 12 17  4  3  5  0   0  0   0   1   0
  7   1 15  5  4  1  0  0  7  1   0  0   2   0   0
  5  10 12 13 10  1  0  0  4  2   1  0   0   0   0
  6   5  7 12  7  0  0  0  1  2   0  0   0   0   0
  3   6  0  3  2  6  1  6  0  1   0  2   0   0   1
  1   0  0  1  0  2  1  2  0  0   0  0   0   0   0
  2   5  0  1  0  3  2  8  1  0   0  0   0   0   0
  8   0  1  0  0  0  0  0  1  0   0  0   0   0   0
  9   0  0  0  0  0  0  0  0  0   0  0   0   0   0
  6.5 0  0  0  0  0  0  0  0  0   0  0   0   0   0
  0   0  0  0  0  0  0  0  0  0   0  0   0   0   0
  8.5 0  0  0  0  0  0  0  0  0   0  0   0   0   0
  4.5 0  0  0  0  0  0  0  0  0   0  0   0   0   0
  3.5 0  0  0  0  0  0  0  0  0   0  0   0   0   0

Overall Statistics

               Accuracy : 0.2925
                 95% CI : (0.243, 0.3458)
    No Information Rate : 0.2327
    P-Value [Acc > NIR] : 0.008135

                  Kappa : 0.1408

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 4 Class: 7 Class: 5 Class: 6 Class: 3 Class: 1
Sensitivity            0.6087  0.29412  0.17568  0.20000  0.20000 0.125000
Specificity            0.6104  0.92135  0.83607  0.90459  0.92361 0.983871
Pos Pred Value         0.3022  0.41667  0.24528  0.20588  0.21429 0.166667
Neg Pred Value         0.8492  0.87234  0.76981  0.90141  0.91724 0.977564
Prevalence             0.2170  0.16038  0.23270  0.11006  0.09434 0.025157
Detection Rate         0.1321  0.04717  0.04088  0.02201  0.01887 0.003145
Detection Prevalence   0.4371  0.11321  0.16667  0.10692  0.08805 0.018868
Balanced Accuracy      0.6096  0.60773  0.50587  0.55230  0.56181 0.554435
                     Class: 2 Class: 8 Class: 9 Class: 6.5 Class: 0 Class: 8.5
Sensitivity           0.42105 0.052632  0.00000   0.000000 0.000000   0.000000
Specificity           0.95987 0.996656  1.00000   1.000000 1.000000   1.000000
Pos Pred Value        0.40000 0.500000      NaN        NaN      NaN        NaN
Neg Pred Value        0.96309 0.943038  0.98113   0.996855 0.993711   0.993711
Prevalence            0.05975 0.059748  0.01887   0.003145 0.006289   0.006289
Detection Rate        0.02516 0.003145  0.00000   0.000000 0.000000   0.000000
Detection Prevalence  0.06289 0.006289  0.00000   0.000000 0.000000   0.000000
Balanced Accuracy     0.69046 0.524644  0.50000   0.500000 0.500000   0.500000
                     Class: 4.5 Class: 3.5
Sensitivity            0.000000   0.000000
Specificity            1.000000   1.000000
```

```
Pos Pred Value              NaN        NaN
Neg Pred Value         0.996855   0.996855
Prevalence             0.003145   0.003145
Detection Rate         0.000000   0.000000
Detection Prevalence   0.000000   0.000000
Balanced Accuracy      0.500000   0.500000
```

```
#optimizing the accuracy
i=1
k.optm=1
set.seed(42)
for (i in 1:75){
  knn.mod <- knn(train=train.loan, test=res$test, cl=train.loan_labels, k=i)
  k.optm[i] <- 100 * sum(test.loan_labels == knn.mod)/NROW(test.loan_labels)
  k=i
  cat(k,'=',k.optm[i],'
      ')
}
```

```
1 = 23.27044
      2 = 23.89937
      3 = 25.78616
      4 = 27.04403
      5 = 26.41509
      6 = 28.30189
      7 = 27.98742
      8 = 28.93082
      9 = 29.24528
      10 = 28.30189
      11 = 27.98742
      12 = 26.72956
      13 = 28.30189
      14 = 27.35849
      15 = 27.04403
      16 = 28.61635
      17 = 25.4717
      18 = 26.72956
      19 = 27.98742
      20 = 27.35849
      21 = 24.5283
      22 = 26.41509
      23 = 26.72956
      24 = 27.98742
      25 = 27.98742
      26 = 26.10063
      27 = 26.72956
      28 = 27.04403
      29 = 29.24528
      30 = 28.93082
      31 = 30.50314
      32 = 29.55975
      33 = 29.55975
      34 = 30.18868
      35 = 29.55975
      36 = 28.30189
      37 = 29.87421
      38 = 28.30189
      39 = 28.61635
      40 = 29.24528
      41 = 29.24528
```

```
      42 = 31.13208
      43 = 29.87421
      44 = 31.76101
      45 = 30.18868
      46 = 30.18868
      47 = 29.24528
      48 = 28.93082
      49 = 28.93082
      50 = 30.18868
      51 = 29.87421
      52 = 30.18868
      53 = 30.18868
      54 = 29.24528
      55 = 29.24528
      56 = 28.30189
      57 = 26.41509
      58 = 29.55975
      59 = 27.04403
      60 = 29.24528
      61 = 28.93082
      62 = 29.55975
      63 = 31.13208
      64 = 29.87421
      65 = 30.18868
      66 = 29.55975
      67 = 30.18868
      68 = 28.61635
      69 = 28.93082
      70 = 29.55975
      71 = 29.55975
      72 = 29.24528
      73 = 30.81761
      74 = 30.50314
      75 = 30.18868


#best is k = 44, acc = 31.76101

knn.44 <- knn(train=train.loan, test=test.loan, cl=train.loan_labels, k=44)
ACC.44 <- 100 * sum(test.loan_labels == knn.44)/NROW(test.loan_labels)
ACC.44

[1] 31.44654

uKNN44 <- union(knn.44, res$test$RPE)
tKNN44 <- table(factor(knn.44, uKNN44), factor(res$test$RPE, uKNN44))
#confusionMatrix(tKNN44)

########################################Test Datasets From Top K of Previous Approaches

########All Possible Regression
allTestLm <- lm(RPE ~ Distance + `HML Distance` + `HML Efforts` + Sprints +
                  Accelerations + Decelerations + `Average Metabolic Power` + `Dynamic Stress Load` +
                  `HML Density` + `Speed Intensity` + Impacts + Duration, data = res$test)
allRegTest <- ols_step_all_possible(allTestLm)
head(allRegTest)

   Index N                Predictors  R-Square Adj. R-Square Mallow's Cp
```

```
2      1 1            ‘HML Distance‘ 0.3653348     0.3633264    98.30825
10     2 1          ‘Speed Intensity‘ 0.2953008     0.2930707   143.80563
7      3 1 ‘Average Metabolic Power‘ 0.2901075     0.2878610   147.17945
1      4 1                   Distance 0.2757790     0.2734871   156.48793
5      5 1             Accelerations 0.2522875     0.2499213   171.74907
3      6 1             ‘HML Efforts‘ 0.2406880     0.2382851   179.28469
```

```
RMSE(allTestLm) #1.258752
```

```
[1] 1.258752
```

```
count <- summary(influence.measures(allTestLm)) #33
```

```
Potentially influential observations of
     lm(formula = RPE ~ Distance + ‘HML Distance‘ + ‘HML Efforts‘ +      Sprints + Accelerations + Deceleratio

     dfb.1_ dfb.Dstn dfb.‘HMLDs‘ dfb.‘HME dfb.Sprn dfb.Accl dfb.Dclr dfb.‘AMP
7    -0.03   0.10     0.02        0.03    -0.02    -0.04     0.03     0.01
8     0.00  -0.02    -0.06        0.06    -0.01     0.00    -0.02     0.00
30    0.05  -0.07     0.16       -0.12    -0.13    -0.14     0.05    -0.04
178   0.26  -0.16     0.01        0.01    -0.15     0.14     0.02    -0.26
194   0.00   0.03    -0.03        0.03     0.02     0.15    -0.11    -0.04
236  -0.04  -0.08    -0.04        0.10    -0.08    -0.01    -0.05    -0.04
369   0.15   0.18     0.07       -0.03     0.14    -0.09    -0.05    -0.11
512  -0.48   0.22     0.03       -0.03     0.05     0.08    -0.17     0.39
527  -0.05   0.04     0.00       -0.06    -0.03    -0.01     0.08     0.01
643   0.02  -0.01     0.01       -0.01    -0.01     0.02    -0.02    -0.01
908   0.02   0.19     0.18       -0.09     0.03    -0.06    -0.06     0.04
1013 -0.01  -0.05    -0.04       -0.08    -0.04    -0.02     0.06    -0.03
1256  0.01  -0.03    -0.06        0.00     0.07    -0.01     0.01    -0.01
1334 -0.02  -0.29    -0.05       -0.01    -0.17    -0.06     0.03     0.17
1382  0.00   0.00     0.00        0.00     0.00     0.00     0.00     0.00
1481  0.87   0.09     0.91       -0.95     0.02     0.13    -0.19    -0.02
1485  0.00  -0.01    -0.01        0.02     0.00    -0.01     0.00    -0.01
1491  0.00   0.00    -0.01        0.01     0.00     0.00     0.00     0.00
1636  0.06  -0.01     0.04       -0.10     0.09    -0.05     0.06     0.10
1640  0.12   0.20     0.11       -0.04     0.02     0.08    -0.05    -0.05
1653 -0.06  -0.01    -0.11        0.03     0.01     0.10    -0.05     0.10
1731 -0.03   0.04     0.03       -0.04    -0.02     0.01     0.04     0.01
1918  0.38  -0.14     0.11       -0.09    -0.01    -0.14     0.02    -0.23
2015  0.01   0.00     0.00        0.00    -0.01     0.00     0.00     0.00
2033 -0.02   0.00    -0.02        0.01     0.02     0.01     0.00     0.04
2157  0.00   0.00    -0.03        0.03     0.02    -0.01     0.01     0.00
2167  0.05  -0.18    -0.54        0.44     0.04    -0.03     0.13    -0.05
2168  0.00   0.00    -0.04        0.03     0.02     0.00     0.00     0.00
2169 -0.04  -0.99    -0.64        0.30    -0.44     0.24    -0.02     0.19
2233 -0.03   0.03    -0.01        0.02     0.06     0.00     0.01     0.05
2249 -0.04   0.06    -0.01        0.03    -0.01    -0.01     0.02     0.02
2272 -0.01   0.02     0.11       -0.20    -0.15    -0.05     0.17    -0.04
2281 -0.09  -0.19    -0.18        0.04    -0.19    -0.22     0.25     0.12
     dfb.‘DSL dfb.‘HMLDn‘ dfb.‘SI‘ dfb.Impc dfb.Drtn dffit   cov.r   cook.d
7     0.10     0.02       -0.10    -0.13    -0.01    -0.24    1.14_*  0.00
8     0.01     0.02        0.01     0.01    -0.01    -0.10    1.16_*  0.00
30   -0.02    -0.06        0.07    -0.02     0.10     0.36    0.83_*  0.01
178   0.00     0.01        0.17     0.00    -0.17     0.51    0.71_*  0.02
194   0.02     0.04       -0.03    -0.02    -0.04     0.17    1.20_*  0.00
236   0.02     0.06        0.07    -0.02     0.18     0.33    0.87_*  0.01
369   0.31    -0.06       -0.18    -0.32    -0.14     0.51    0.68_*  0.02
512   0.03     0.11       -0.23    -0.04     0.43     0.63_*  1.10    0.03
```

```
527   0.05     0.04       -0.03    -0.05     0.01     0.16    1.13_*  0.00
643   0.00    -0.03        0.01     0.00     0.00    -0.04    1.14_*  0.00
908   0.02    -0.04       -0.20     0.02    -0.01    -0.31    1.28_*  0.01
1013 -0.07    -0.02        0.08     0.02    -0.04     0.29    1.17_*  0.01
1256  0.03    -0.01        0.04    -0.04    -0.02     0.17    1.49_*  0.00
1334 -1.00    -0.02        0.31     0.77     0.14    -1.09_*  1.27_*  0.09
1382  0.00     0.00        0.00     0.00     0.00    -0.01    1.16_*  0.00
1481 -0.25    -1.92_*     -0.07     0.18    -0.16    -2.07_*  1.25_*  0.32
1485 -0.01     0.03        0.01     0.01    -0.01     0.05    1.15_*  0.00
1491  0.00     0.02        0.00     0.00     0.00     0.02    1.38_*  0.00
1636  0.02    -0.08       -0.01     0.02     0.04    -0.36    1.20_*  0.01
1640  0.18    -0.07       -0.19    -0.21    -0.36    -0.54    0.85_*  0.02
1653 -0.05     0.06        0.01     0.04     0.02    -0.27    1.20_*  0.01
1731  0.01     0.00       -0.03     0.00     0.02     0.10    1.18_*  0.00
1918 -0.13    -0.24        0.15     0.10    -0.03     0.52    0.83_*  0.02
2015  0.00     0.00        0.00     0.00     0.00    -0.02    1.14_*  0.00
2033  0.01     0.01        0.00    -0.01     0.01     0.05    1.14_*  0.00
2157  0.00     0.01        0.00     0.00     0.00    -0.05    1.20_*  0.00
2167  0.05     0.06        0.18    -0.02    -0.02    -0.89_*  1.28_*  0.06
2168 -0.01     0.01        0.00     0.01     0.00    -0.07    1.32_*  0.00
2169  0.42     0.03        0.97    -0.35     0.23     1.40_*  1.06    0.15
2233  0.04     0.01       -0.03    -0.05     0.02     0.09    1.17_*  0.00
2249  0.03     0.04       -0.06    -0.02     0.00    -0.12    1.14_*  0.00
2272 -0.08     0.00       -0.01     0.05     0.04     0.38    0.84_*  0.01
2281  0.08     0.12        0.17    -0.06     0.16    -0.50    0.84_*  0.02
     hat
7     0.10
8     0.11
30    0.02
178   0.03
194   0.14_*
236   0.02
369   0.02
512   0.14_*
527   0.09
643   0.09
908   0.20_*
1013  0.13_*
1256  0.30_*
1334  0.28_*
1382  0.10
1481  0.38_*
1485  0.10
1491  0.24_*
1636  0.16_*
1640  0.05
1653  0.15_*
1731  0.12
1918  0.04
2015  0.08
2033  0.09
2157  0.13_*
2167  0.26_*
2168  0.21_*
2169  0.24_*
2233  0.11
2249  0.09
2272  0.03
2281  0.04
```

```
outlierTest(allTestLm)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
369 3.272784          0.0011876      0.37767
```

```
########Forward AIC
forwardTestlm <- lm(RPE ~ `Average Metabolic Power` + Decelerations + `HML Density` +
                      `Dynamic Stress Load` + Impacts + `Max Speed` + Accelerations +
                      `HML Efforts` + Distance + Duration + `Speed Intensity` + `HML Distance`, data = res$tes
forwardTest <- ols_step_forward_aic(forwardTestlm) #RPE ~ , data = res$test
forwardTest
```

```
                              Selection Summary
--------------------------------------------------------------------------------
Variable                     AIC       Sum Sq       RSS       R-Sq      Adj. R-Sq
--------------------------------------------------------------------------------
‘HML Distance‘           1150.666    392.082    681.130    0.36533      0.36333
‘HML Density‘            1136.516    425.810    647.402    0.39676      0.39293
Decelerations            1114.116    473.627    599.584    0.44132      0.43598
‘Dynamic Stress Load‘    1099.247    504.604    568.607    0.47018      0.46341
Duration                 1093.231    518.759    554.453    0.48337      0.47509
Distance                 1087.181    532.617    540.594    0.49628      0.48657
Accelerations            1082.391    544.039    529.172    0.50693      0.49579
‘Speed Intensity‘        1078.960    552.999    520.212    0.51528      0.50273
‘Max Speed‘              1078.938    556.297    516.915    0.51835      0.50427
--------------------------------------------------------------------------------
```

```
RMSE(forwardTestlm) #1.274529
```

```
[1] 1.274529
```

```
count1 <- summary(influence.measures(forwardTestlm)) #32
```

```
Potentially influential observations of
     lm(formula = RPE ~ ‘Average Metabolic Power‘ + Decelerations +      ‘HML Density‘ + ‘Dynamic Stress Load‘

     dfb.1_ dfb.‘AMP dfb.Dclr dfb.‘HMLDn‘ dfb.‘DSL dfb.Impc dfb.‘MS‘ dfb.Accl
7     -0.06   0.01     0.03     0.02        0.13    -0.16     0.05    -0.05
8      0.00   0.01    -0.03     0.03        0.01     0.01     0.00     0.01
30     0.06  -0.03     0.02    -0.05        0.00    -0.04    -0.02    -0.14
178    0.21  -0.26    -0.01     0.02        0.02    -0.03     0.00     0.16
194    0.01  -0.01    -0.04     0.02        0.01    -0.01    -0.01     0.06
197    0.21  -0.10     0.02    -0.01       -0.06     0.07    -0.11     0.04
236    0.05  -0.02    -0.07     0.08        0.03    -0.03    -0.12     0.00
369    0.08  -0.11    -0.02    -0.07        0.28    -0.29     0.04    -0.09
512   -0.38   0.36    -0.15     0.10        0.02    -0.04     0.05     0.07
643    0.03  -0.02    -0.05    -0.06        0.00     0.00    -0.01     0.05
908   -0.05   0.02    -0.05    -0.06        0.02     0.02     0.11    -0.07
1013  -0.01  -0.03     0.06    -0.03       -0.08     0.02     0.00    -0.02
1256  -0.07   0.02    -0.06     0.02       -0.06     0.06     0.10     0.05
1334   0.08   0.21    -0.01     0.01       -1.08_*   0.84    -0.14    -0.05
1481   0.57  -0.04    -0.18    -1.93_*     -0.25     0.18     0.14     0.13
1485   0.00  -0.01     0.00     0.04       -0.01     0.01     0.00    -0.01
1491   0.01   0.01     0.00    -0.05        0.00     0.00     0.00     0.00
1564   0.05   0.03    -0.01     0.03        0.04    -0.02    -0.10     0.03
1629   0.01   0.01     0.00     0.00       -0.01     0.00    -0.03    -0.01
```

```
1636 -0.01   0.07     0.06    -0.08        0.01     0.03     0.07    -0.05
1653 -0.09   0.11    -0.06     0.05       -0.06     0.04     0.06     0.11
1727 -0.14  -0.03     0.01    -0.02        0.03    -0.04     0.18     0.01
1731 -0.03   0.01     0.05     0.01        0.01    -0.01    -0.02     0.01
2015  0.02   0.00    -0.01     0.00        0.01     0.00    -0.01    -0.01
2033  0.02  -0.02    -0.01     0.00        0.00     0.00    -0.01     0.00
2157  0.01   0.00    -0.01    -0.01        0.01    -0.01    -0.01     0.01
2167 -0.04  -0.07     0.14     0.04        0.05    -0.03     0.12    -0.04
2168  0.02   0.00    -0.02    -0.03        0.03    -0.03    -0.02     0.01
2169  0.16   0.26    -0.13     0.11        0.53    -0.46    -0.28     0.32
2249 -0.06   0.02     0.02     0.05        0.04    -0.04     0.03    -0.01
2272  0.04  -0.03     0.15     0.03       -0.07     0.03    -0.07    -0.04
2281 -0.09   0.13     0.24     0.14        0.11    -0.11     0.03    -0.23
     dfb.'HME dfb.Dstn dfb.Drtn dfb.'SI' dfb.'HMLDs' dffit   cov.r   cook.d
7     0.03     0.16    -0.02    -0.16     0.02       -0.28    1.13_*  0.01
8     0.09    -0.01    -0.01     0.01    -0.08       -0.14    1.16_*  0.00
30   -0.10    -0.01     0.09     0.01     0.15        0.35    0.81_*  0.01
178   0.04    -0.10    -0.19     0.10     0.00        0.50    0.69_*  0.02
194   0.01     0.01    -0.01    -0.01    -0.01        0.06    1.22_*  0.00
197   0.00    -0.08    -0.09     0.08     0.04        0.33    0.86_*  0.01
236   0.13    -0.08     0.20     0.07    -0.06        0.34    0.87_*  0.01
369  -0.06     0.13    -0.12    -0.12     0.08        0.47    0.71_*  0.02
512  -0.05     0.22     0.39    -0.23     0.03        0.60    1.11    0.03
643  -0.01    -0.02    -0.01     0.02     0.02       -0.10    1.13_*  0.00
908  -0.12     0.23    -0.03    -0.25     0.21       -0.36    1.31_*  0.01
1013 -0.08    -0.05    -0.05     0.08    -0.05        0.33    1.16_*  0.01
1256  0.01     0.21     0.00    -0.24     0.15       -0.43    1.38_*  0.01
1334  0.05    -0.29     0.16     0.32    -0.09       -1.20_*  1.23_*  0.11
1481 -0.99     0.12    -0.19    -0.10     0.93       -2.09_*  1.24_*  0.33
1485  0.02    -0.01    -0.01     0.01    -0.02        0.06    1.15_*  0.00
1491 -0.03     0.00     0.00     0.00     0.03       -0.06    1.38_*  0.00
1564  0.03     0.03     0.02    -0.04     0.00       -0.14    1.21_*  0.00
1629  0.00    -0.01     0.04     0.01     0.00        0.06    1.13_*  0.00
1636 -0.10    -0.03     0.02     0.01     0.05       -0.29    1.21_*  0.01
1653  0.02     0.00     0.01     0.00    -0.12       -0.31    1.20_*  0.01
1727 -0.03     0.01     0.02    -0.02     0.01        0.20    1.24_*  0.00
1731 -0.06     0.07     0.03    -0.07     0.03        0.14    1.18_*  0.00
2015  0.02    -0.01     0.01     0.00     0.01       -0.09    1.13_*  0.00
2033  0.00     0.01    -0.01     0.00     0.01       -0.03    1.14_*  0.00
2157 -0.03     0.01     0.00    -0.01     0.04        0.06    1.18_*  0.00
2167  0.42    -0.19    -0.05     0.20    -0.52       -0.91_*  1.29_*  0.06
2168 -0.08     0.03     0.00    -0.03     0.10        0.16    1.28_*  0.00
2169  0.47    -1.05_*   0.25     1.03_*  -0.79        1.53_*  0.96    0.18
2249  0.03     0.10     0.00    -0.10    -0.01       -0.17    1.14_*  0.00
2272 -0.17     0.09     0.04    -0.08     0.09        0.38    0.82_*  0.01
2281  0.08    -0.11     0.14     0.09    -0.20       -0.51    0.80_*  0.02
     hat
7     0.10
8     0.10
30    0.02
178   0.02
194   0.15_*
197   0.02
236   0.02
369   0.02
512   0.14_*
643   0.08
908   0.22_*
1013  0.13_*
1256  0.26_*
```

```
1334  0.28_*
1481  0.38_*
1485  0.10
1491  0.24_*
1564  0.14_*
1629  0.08
1636  0.16_*
1653  0.15_*
1727  0.17_*
1731  0.12
2015  0.08
2033  0.08
2157  0.12
2167  0.26_*
2168  0.19_*
2169  0.22_*
2249  0.09
2272  0.02
2281  0.03
```

```
outlierTest(forwardTestlm)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
178 3.246876          0.0012972       0.4125
```

```
########Backward AIC
backwardTestLm <- lm(RPE ~ `Average Metabolic Power`  + `HML Density` + Decelerations + Accelerations +
                       `Dynamic Stress Load` + Distance + `HML Distance` + Duration + Impacts +
                       `HML Efforts` + `Speed Intensity`, data = res$test)
backwardTest <- ols_step_backward_aic(lmTest) #RPE ~ , data = res$test
backwardTest
```

```
                          Backward Elimination Summary
-------------------------------------------------------------------------------------
Variable                        AIC         RSS       Sum Sq      R-Sq      Adj. R-Sq
-------------------------------------------------------------------------------------
Full Model                  1079.098    498.023    575.188    0.53595      0.51290
‘Speed Intensity‘           1077.127    498.068    575.143    0.53591      0.51447
Impacts                     1075.280    498.308    574.904    0.53569      0.51583
‘Average Metabolic Power‘   1073.550    498.731    574.481    0.53529      0.51701
‘HML Efforts‘               1072.025    499.477    573.735    0.53460      0.51787
‘Max Speed‘                 1070.867    500.801    572.410    0.53336      0.51816
‘Sprint Distance‘           1069.914    502.453    570.758    0.53182      0.51814
-------------------------------------------------------------------------------------
```

```
RMSE(backwardTestLm) #1.277867
```

```
[1] 1.277867
```

```
count10 <- summary(influence.measures(backwardTestLm)) #32
```

```
Potentially influential observations of
     lm(formula = RPE ~ ‘Average Metabolic Power‘ + ‘HML Density‘ +      Decelerations + Accelerations + ‘Dynar

     dfb.1_ dfb.‘AMP dfb.‘HMLDn‘ dfb.Dclr dfb.Accl dfb.‘DSL dfb.Dstn
7    -0.03   0.01     0.03        0.03    -0.04     0.11     0.14
```

8	-0.01	0.01	0.03	-0.03	0.01	0.01	-0.02
30	0.06	-0.04	-0.05	0.02	-0.14	0.00	0.00
178	0.28	-0.27	0.02	-0.01	0.16	0.02	-0.11
194	0.00	-0.03	0.03	-0.07	0.10	0.01	0.02
197	0.18	-0.12	-0.03	0.02	0.04	-0.06	-0.06
236	-0.04	-0.04	0.06	-0.07	0.00	0.03	-0.05
252	0.05	0.00	0.00	-0.04	-0.09	-0.03	-0.06
369	0.14	-0.11	-0.06	-0.02	-0.09	0.28	0.12
512	-0.45	0.36	0.10	-0.15	0.07	0.02	0.20
643	0.04	-0.02	-0.07	-0.05	0.05	0.00	-0.02
696	-0.18	0.17	0.04	-0.09	0.10	-0.06	0.02
908	0.02	0.03	-0.03	-0.04	-0.04	0.01	0.14
1013	-0.01	-0.03	-0.03	0.06	-0.02	-0.08	-0.05
1256	-0.01	0.02	0.03	-0.05	0.04	-0.05	0.15
1334	-0.02	0.20	-0.01	0.00	-0.06	-1.12_*	-0.28
1481	0.85	-0.01	-1.88_*	-0.18	0.13	-0.25	0.09
1485	0.00	-0.01	0.04	0.00	-0.01	-0.01	-0.01
1491	0.02	0.01	-0.07	0.00	0.00	0.00	0.00
1636	0.04	0.06	-0.05	0.05	-0.03	0.01	-0.04
1640	0.12	-0.05	-0.07	-0.05	0.08	0.18	0.22
1653	-0.06	0.10	0.05	-0.05	0.10	-0.05	-0.01
1731	-0.06	0.01	0.01	0.06	0.01	0.01	0.09
1918	0.38	-0.22	-0.24	0.02	-0.13	-0.13	-0.16
2015	0.02	-0.01	0.00	-0.01	-0.01	0.01	-0.01
2157	0.01	0.00	-0.01	-0.02	0.01	0.01	0.02
2167	0.04	-0.05	0.05	0.13	-0.03	0.04	-0.21
2168	0.01	0.00	-0.04	-0.03	0.02	0.03	0.04
2169	-0.03	0.23	0.06	-0.12	0.33	0.56	-1.05_*
2249	-0.04	0.02	0.05	0.02	-0.01	0.03	0.08
2272	-0.01	-0.04	0.01	0.16	-0.04	-0.07	0.11
2281	-0.09	0.13	0.14	0.23	-0.22	0.11	-0.12

	dfb.‘HMLDs‘	dfb.Drtn	dfb.Impc	dfb.‘HME	dfb.‘SI‘	dffit	cov.r	cook.d
7	0.02	-0.01	-0.15	0.03	-0.14	-0.26	1.13_*	0.01
8	-0.08	-0.01	0.01	0.09	0.01	-0.14	1.15_*	0.00
30	0.16	0.08	-0.04	-0.10	0.00	0.35	0.82_*	0.01
178	0.00	-0.19	-0.03	0.04	0.10	0.50	0.71_*	0.02
194	-0.02	-0.02	-0.01	0.02	-0.02	0.11	1.20_*	0.00
197	0.05	-0.12	0.06	-0.02	0.05	0.32	0.86_*	0.01
236	-0.05	0.18	-0.04	0.12	0.04	0.33	0.87_*	0.01
252	-0.02	-0.03	0.03	0.06	0.06	-0.18	1.12_*	0.00
369	0.08	-0.12	-0.28	-0.06	-0.12	0.47	0.74_*	0.02
512	0.03	0.40	-0.03	-0.04	-0.22	0.58	1.12_*	0.03
643	0.02	-0.02	0.00	-0.01	0.02	-0.11	1.13_*	0.00
696	-0.05	0.12	0.07	0.00	-0.02	-0.27	0.87_*	0.01
908	0.13	0.00	0.02	-0.07	-0.15	-0.23	1.28_*	0.00
1013	-0.06	-0.05	0.02	-0.08	0.08	0.34	1.16_*	0.01
1256	0.10	0.02	0.05	0.02	-0.17	-0.32	1.36_*	0.01
1334	-0.08	0.14	0.86	0.03	0.30	-1.24_*	1.22_*	0.13
1481	0.89	-0.15	0.18	-0.95	-0.06	-2.02_*	1.29_*	0.34
1485	-0.02	-0.01	0.01	0.02	0.01	0.05	1.15_*	0.00
1491	0.04	0.00	0.00	-0.04	-0.01	-0.07	1.37_*	0.00
1636	0.03	0.03	0.03	-0.07	0.03	-0.22	1.21_*	0.00
1640	0.10	-0.36	-0.21	-0.04	-0.21	-0.53	0.88_*	0.02
1653	-0.11	0.02	0.04	0.03	0.01	-0.26	1.20_*	0.01
1731	0.04	0.03	-0.01	-0.07	-0.08	0.16	1.17_*	0.00
1918	0.11	-0.03	0.10	-0.09	0.16	0.51	0.85_*	0.02
2015	0.01	0.01	0.00	0.03	0.00	-0.09	1.12_*	0.00
2157	0.05	0.00	-0.01	-0.04	-0.02	0.08	1.18_*	0.00
2167	-0.50	-0.02	-0.02	0.41	0.21	-0.84_*	1.30_*	0.06
2168	0.12	0.00	-0.04	-0.10	-0.05	0.19	1.27_*	0.00

```
2169 -0.79        0.20    -0.51     0.45     1.02_*   1.55_*  0.95    0.20
2249 -0.01        0.00    -0.03     0.03    -0.08    -0.14    1.13_*  0.00
2272  0.11        0.02     0.02    -0.18    -0.10     0.37    0.83_*  0.01
2281 -0.21        0.15    -0.11     0.08     0.10    -0.50    0.82_*  0.02
     hat
7     0.10
8     0.10
30    0.02
178   0.02
194   0.14_*
197   0.02
236   0.02
252   0.08
369   0.02
512   0.14_*
643   0.08
696   0.01
908   0.20_*
1013  0.13_*
1256  0.25_*
1334  0.27_*
1481  0.38_*
1485  0.10
1491  0.24_*
1636  0.15_*
1640  0.05
1653  0.15_*
1731  0.12_*
1918  0.04
2015  0.08
2157  0.12_*
2167  0.26_*
2168  0.19_*
2169  0.22_*
2249  0.09
2272  0.02
2281  0.03
```

```
outlierTest(backwardTestLm)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
178 3.243576          0.0013113        0.417
```

```
########Stepwise AIC
stepwiseTest0 <- lm(RPE ~ 1, data = res$test) #empty modfit
stepwiseTestAll <- lm(RPE ~ `Average Metabolic Power`  + `HML Density` + Decelerations + Accelerations +
                        `Dynamic Stress Load` + Distance + `HML Distance` + Duration + Impacts +
                        `HML Efforts` + `Speed Intensity`, data = res$test) #full modfit , add the variables h
stepwiseTestBest <- stepAIC(stepwiseTestAll, direction="both",
                            scope=list(upper=stepwiseTestAll, lower=stepwiseTest0) ) #finds best modfit
```

```
Start:  AIC=179.94
RPE ~ `Average Metabolic Power` + `HML Density` + Decelerations +
    Accelerations + `Dynamic Stress Load` + Distance + `HML Distance` +
    Duration + Impacts + `HML Efforts` + `Speed Intensity`

                            Df Sum of Sq    RSS    AIC
```

```
- Impacts                    1    0.0824 519.36 177.99
- `Average Metabolic Power`  1    0.3538 519.63 178.16
- `HML Efforts`              1    0.5721 519.85 178.29
- Decelerations              1    1.5863 520.86 178.91
<none>                                  519.28 179.94
- `HML Distance`             1    5.0545 524.33 181.02
- `Dynamic Stress Load`      1    5.7960 525.07 181.47
- `Speed Intensity`          1    8.9280 528.20 183.36
- Distance                   1   11.6689 530.95 185.01
- Accelerations              1   14.6499 533.93 186.79
- Duration                   1   21.8831 541.16 191.07
- `HML Density`              1   27.7502 547.03 194.50

Step:  AIC=177.99
RPE ~ `Average Metabolic Power` + `HML Density` + Decelerations +
    Accelerations + `Dynamic Stress Load` + Distance + `HML Distance` +
    Duration + `HML Efforts` + `Speed Intensity`

                           Df Sum of Sq    RSS    AIC
- `Average Metabolic Power`  1    0.3464 519.70 176.21
- `HML Efforts`              1    0.5670 519.93 176.34
- Decelerations              1    1.5040 520.86 176.91
<none>                                  519.36 177.99
- `HML Distance`             1    5.0584 524.42 179.07
+ Impacts                    1    0.0824 519.28 179.94
- `Speed Intensity`          1    8.9392 528.30 181.42
- Distance                   1   11.6673 531.03 183.06
- Accelerations              1   14.5727 533.93 184.79
- Duration                   1   22.1968 541.56 189.30
- `Dynamic Stress Load`      1   26.5205 545.88 191.83
- `HML Density`              1   27.6685 547.03 192.50

Step:  AIC=176.2
RPE ~ `HML Density` + Decelerations + Accelerations + `Dynamic Stress Load` +
    Distance + `HML Distance` + Duration + `HML Efforts` + `Speed Intensity`

                           Df Sum of Sq    RSS    AIC
- `HML Efforts`              1    0.5075 520.21 174.51
- Decelerations              1    1.6585 521.36 175.22
<none>                                  519.70 176.21
- `HML Distance`             1    4.7509 524.46 177.10
+ `Average Metabolic Power`  1    0.3464 519.36 177.99
+ Impacts                    1    0.0749 519.63 178.16
- `Speed Intensity`          1    9.1533 528.86 179.76
- Distance                   1   11.5974 531.30 181.22
- Accelerations              1   14.2305 533.94 182.79
- `Dynamic Stress Load`      1   26.2814 545.99 189.89
- `HML Density`              1   29.1313 548.84 191.55
- Duration                   1   30.2931 550.00 192.22

Step:  AIC=174.52
RPE ~ `HML Density` + Decelerations + Accelerations + `Dynamic Stress Load` +
    Distance + `HML Distance` + Duration + `Speed Intensity`

                           Df Sum of Sq    RSS    AIC
- Decelerations              1     1.407 521.62 173.37
<none>                                  520.21 174.51
+ `HML Efforts`              1     0.507 519.70 176.21
+ `Average Metabolic Power`  1     0.287 519.93 176.34
+ Impacts                    1     0.071 520.14 176.47
```

```
- ‘HML Distance‘             1     7.517 527.73 177.08
- ‘Speed Intensity‘          1     8.960 529.17 177.95
- Distance                   1    11.838 532.05 179.67
- Accelerations              1    14.375 534.59 181.18
- ‘Dynamic Stress Load‘      1    26.467 546.68 188.30
- Duration                   1    29.881 550.09 190.28
- ‘HML Density‘              1    62.356 582.57 208.52

Step:  AIC=173.37
RPE ~ ‘HML Density‘ + Accelerations + ‘Dynamic Stress Load‘ +
    Distance + ‘HML Distance‘ + Duration + ‘Speed Intensity‘

                           Df Sum of Sq    RSS    AIC
<none>                                   521.62 173.37
+ Decelerations             1     1.407 520.21 174.51
+ ‘Average Metabolic Power‘  1     0.436 521.18 175.11
+ ‘HML Efforts‘             1     0.256 521.36 175.22
+ Impacts                   1     0.000 521.62 175.37
- ‘HML Distance‘             1     9.870 531.49 177.34
- ‘Speed Intensity‘          1    12.932 534.55 179.16
- Distance                   1    17.109 538.73 181.64
- ‘Dynamic Stress Load‘      1    27.432 549.05 187.67
- Duration                   1    31.433 553.05 189.98
- Accelerations              1    36.571 558.19 192.92
- ‘HML Density‘              1    60.977 582.60 206.53

formula(stepwiseTestBest)

RPE ~ ‘HML Density‘ + Accelerations + ‘Dynamic Stress Load‘ +
    Distance + ‘HML Distance‘ + Duration + ‘Speed Intensity‘

summary(stepwiseTestBest)

Call:
lm(formula = RPE ~ ‘HML Density‘ + Accelerations + ‘Dynamic Stress Load‘ +
    Distance + ‘HML Distance‘ + Duration + ‘Speed Intensity‘,
    data = res$test)

Residuals:
    Min      1Q  Median      3Q     Max
-3.4403 -0.8816 -0.0804  0.8424  4.1585

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)           1.9525207  0.3445907   5.666 3.34e-08 ***
‘HML Density‘         0.1165195  0.0193559   6.020 4.93e-09 ***
Accelerations         0.0232944  0.0049967   4.662 4.66e-06 ***
‘Dynamic Stress Load‘ 0.0034734  0.0008602   4.038 6.81e-05 ***
Distance              0.0018551  0.0005818   3.189  0.00157 **
‘HML Distance‘        0.0007855  0.0003243   2.422  0.01601 *
Duration             -0.0110194  0.0025495  -4.322 2.08e-05 ***
‘Speed Intensity‘    -0.0348705  0.0125782  -2.772  0.00590 **
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

Residual standard error: 1.297 on 310 degrees of freedom
Multiple R-squared:  0.514, Adjusted R-squared:  0.503
F-statistic: 46.83 on 7 and 310 DF,  p-value: < 2.2e-16

coefficients(stepwiseTestBest)
```

```
          (Intercept)         `HML Density`         Accelerations
          1.952520741           0.116519507           0.023294352
`Dynamic Stress Load`              Distance        `HML Distance`
          0.003473416           0.001855085           0.000785524
             Duration     `Speed Intensity`
         -0.011019449          -0.034870466
```

```
RMSE(stepwiseTestBest) #1.280747
```

```
[1] 1.280747
```

```
count2 <- summary(influence.measures(stepwiseTestBest)) #41
```

```
Potentially influential observations of
     lm(formula = RPE ~ `HML Density` + Accelerations + `Dynamic Stress Load` +      Distance + `HML Distance`

     dfb.1_ dfb.`HMLDn` dfb.Accl dfb.`DSL dfb.Dstn dfb.`HMLDs` dfb.Drtn
7      0.01  -0.01       -0.07    -0.04     0.15     0.08       -0.05
8      0.02  -0.02        0.00     0.02     0.01    -0.01       -0.02
30     0.03   0.02       -0.20    -0.10    -0.07     0.12        0.15
178    0.16   0.02        0.18    -0.02    -0.10     0.01       -0.06
194    0.00   0.01        0.02     0.00     0.00    -0.01        0.00
197    0.13  -0.01        0.08    -0.03    -0.05     0.06       -0.06
236   -0.05  -0.01       -0.06    -0.03    -0.03     0.06        0.24
252    0.07  -0.04       -0.11    -0.01    -0.04     0.03       -0.04
297    0.02   0.01       -0.12    -0.02    -0.01     0.00       -0.01
347    0.00   0.08       -0.12     0.50     0.15     0.06       -0.05
369    0.15  -0.03       -0.25     0.05     0.03     0.02       -0.08
377    0.00   0.00        0.00     0.00     0.00     0.00        0.00
497   -0.27   0.10       -0.11    -0.07     0.04     0.06        0.43
500   -0.10   0.04       -0.04     0.04     0.05     0.04        0.11
511    0.16  -0.07       -0.05    -0.01     0.00     0.08       -0.14
643    0.03  -0.06        0.01     0.01    -0.03     0.01        0.00
696   -0.10   0.06        0.08     0.01    -0.01    -0.08        0.04
908    0.02   0.02       -0.08     0.06     0.10     0.12        0.00
1013  -0.09   0.04        0.03    -0.16    -0.08    -0.24       -0.04
1256   0.01   0.02        0.03     0.00     0.15     0.16        0.01
1263   0.09  -0.07        0.09    -0.39     0.10     0.16       -0.19
1334  -0.03  -0.02        0.18    -0.65    -0.07    -0.03        0.08
1382   0.03   0.01        0.01     0.02    -0.02    -0.06       -0.01
1481   0.83  -1.48_*     -0.04    -0.18    -0.42     0.13        0.04
1485  -0.01   0.03        0.00    -0.01     0.00     0.00       -0.01
1491   0.05  -0.09        0.00     0.01    -0.03     0.01        0.01
1629  -0.01   0.00       -0.02    -0.02    -0.02     0.00        0.08
1632   0.00   0.00        0.01     0.01     0.00     0.00       -0.02
1636   0.09  -0.02        0.00     0.09    -0.07    -0.03        0.01
1653   0.01   0.03        0.08    -0.02    -0.01    -0.10       -0.03
1731  -0.13   0.09        0.09     0.02     0.11    -0.02        0.04
1918   0.29  -0.22       -0.17    -0.11    -0.20     0.03        0.16
2015   0.03  -0.02       -0.02     0.01     0.00     0.05        0.01
2153  -0.01   0.08        0.01    -0.01    -0.07    -0.03        0.02
2157   0.00   0.01       -0.01     0.00    -0.01     0.02        0.00
2167   0.12  -0.29        0.10     0.07     0.03    -0.24       -0.08
2168   0.00   0.04       -0.03     0.00    -0.03     0.06        0.01
2169   0.41  -0.33        0.22     0.19    -1.09_*  -0.69       -0.04
2249  -0.03   0.03        0.00     0.01     0.10     0.03       -0.02
2272  -0.13   0.18        0.07    -0.11     0.10    -0.05        0.09
2281   0.04   0.07       -0.10     0.06    -0.04    -0.17        0.04
     dfb.`SI` dffit   cov.r   cook.d hat
```

```
7    -0.15    -0.20    1.08_*  0.01   0.07
8    -0.01    -0.06    1.09_*  0.00   0.06
30    0.06     0.34    0.88_*  0.01   0.02
178   0.08     0.42    0.79_*  0.02   0.02
194   0.00     0.02    1.10_*  0.00   0.07
197   0.03     0.29    0.90_*  0.01   0.01
236   0.02     0.29    0.92_*  0.01   0.02
252   0.04    -0.14    1.10_*  0.00   0.07
297   0.01    -0.14    1.09_*  0.00   0.07
347  -0.16     0.54_*  0.97    0.04   0.06
369  -0.02     0.34    0.81_*  0.01   0.01
377   0.00     0.00    1.09_*  0.00   0.06
497  -0.04     0.51_*  0.99    0.03   0.06
500  -0.05     0.17    1.09_*  0.00   0.07
511  -0.01    -0.22    1.08_*  0.01   0.07
643   0.03    -0.07    1.09_*  0.00   0.06
696   0.01    -0.17    0.92_*  0.00   0.01
908  -0.12    -0.18    1.22_*  0.00   0.16_*
1013  0.12     0.36    1.13_*  0.02   0.12_*
1256 -0.16    -0.24    1.33_*  0.01   0.23_*
1263 -0.09    -0.55_*  0.98    0.04   0.06
1334  0.09    -0.69_*  1.08_*  0.06   0.13_*
1382  0.02    -0.13    1.08_*  0.00   0.06
1481  0.39    -1.57_*  1.22_*  0.30   0.28_*
1485  0.00     0.04    1.11_*  0.00   0.08_*
1491  0.03    -0.10    1.22_*  0.00   0.16_*
1629  0.02     0.10    1.09_*  0.00   0.06
1632  0.00    -0.02    1.08_*  0.00   0.05
1636  0.06    -0.21    1.15_*  0.01   0.11_*
1653  0.01    -0.16    1.15_*  0.00   0.11_*
1731 -0.11     0.19    1.11_*  0.00   0.08_*
1918  0.18     0.45    0.89_*  0.02   0.03
2015 -0.01    -0.07    1.10_*  0.00   0.07
2153  0.07     0.16    1.08_*  0.00   0.06
2157  0.01     0.05    1.11_*  0.00   0.07
2167 -0.01    -0.62_*  1.19_*  0.05   0.18_*
2168  0.02     0.15    1.16_*  0.00   0.12_*
2169   1.09_*   1.23_*  1.02    0.18   0.17_*
2249 -0.10    -0.12    1.11_*  0.00   0.08_*
2272 -0.10     0.30    0.87_*  0.01   0.01
2281  0.04    -0.42    0.87_*  0.02   0.02
```

```
outlierTest(stepwiseTestBest)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
178 3.282641          0.0011463      0.36451
```

```
########Normal GLM
summary(glmTest <- glm(RPE ~ `Average Metabolic Power`  + `HML Density` + Decelerations + Accelerations +
                         `Dynamic Stress Load` + Distance + `HML Distance` + Duration + Impacts +
                         `HML Efforts` + `Speed Intensity`, data = res$test))
```

```
Call:
glm(formula = RPE ~ `Average Metabolic Power` + `HML Density` +
    Decelerations + Accelerations + `Dynamic Stress Load` + Distance +
    `HML Distance` + Duration + Impacts + `HML Efforts` + `Speed Intensity`,
    data = res$test)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.3821  -0.8849  -0.0782   0.8642   4.1125

Coefficients:
                           Estimate Std. Error t value Pr(>|t|)
(Intercept)                2.1771373  0.5114160   4.257 2.76e-05 ***
‘Average Metabolic Power‘ -0.0532045  0.1165203  -0.457 0.648275
‘HML Density‘              0.1077887  0.0266550   4.044 6.66e-05 ***
Decelerations              0.0070293  0.0072704   0.967 0.334390
Accelerations              0.0197952  0.0067372   2.938 0.003552 **
‘Dynamic Stress Load‘      0.0038452  0.0020806   1.848 0.065553 .
Distance                   0.0018498  0.0007054   2.622 0.009172 **
‘HML Distance‘             0.0010101  0.0005853   1.726 0.085386 .
Duration                  -0.0119213  0.0033198  -3.591 0.000384 ***
Impacts                   -0.0010719  0.0048652  -0.220 0.825772
‘HML Efforts‘             -0.0026254  0.0045218  -0.581 0.561931
‘Speed Intensity‘         -0.0343346  0.0149690  -2.294 0.022484 *
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

(Dispersion parameter for gaussian family taken to be 1.696981)

    Null deviance: 1073.21  on 317  degrees of freedom
Residual deviance:  519.28  on 306  degrees of freedom
AIC: 1084.4

Number of Fisher Scoring iterations: 2

RMSE(glmTest) #1.277867

[1] 1.277867

count3 <- summary(influence.measures(glmTest)) #32

Potentially influential observations of
     glm(formula = RPE ~ ‘Average Metabolic Power‘ + ‘HML Density‘ +      Decelerations + Accelerations + ‘Dyn

     dfb.1_ dfb.‘AMP dfb.‘HMLDn‘ dfb.Dclr dfb.Accl dfb.‘DSL dfb.Dstn
7     -0.03    0.01      0.03       0.03    -0.04     0.11     0.14
8     -0.01    0.01      0.03      -0.03     0.01     0.01    -0.02
30     0.06   -0.04     -0.05       0.02    -0.14     0.00     0.00
178    0.28   -0.27      0.02      -0.01     0.16     0.02    -0.11
194    0.00   -0.03      0.03      -0.07     0.10     0.01     0.02
197    0.18   -0.12     -0.03       0.02     0.04    -0.06    -0.06
236   -0.04   -0.04      0.06      -0.07     0.00     0.03    -0.05
252    0.05    0.00      0.00      -0.04    -0.09    -0.03    -0.06
369    0.14   -0.11     -0.06      -0.02    -0.09     0.28     0.12
512   -0.45    0.36      0.10      -0.15     0.07     0.02     0.20
643    0.04   -0.02     -0.07      -0.05     0.05     0.00    -0.02
696   -0.18    0.17      0.04      -0.09     0.10    -0.06     0.02
908    0.02    0.03     -0.03      -0.04    -0.04     0.01     0.14
1013  -0.01   -0.03     -0.03       0.06    -0.02    -0.08    -0.05
1256  -0.01    0.02      0.03      -0.05     0.04    -0.05     0.15
1334  -0.02    0.20     -0.01       0.00    -0.06    -1.12_*  -0.28
1481   0.85   -0.01     -1.88_*    -0.18     0.13    -0.25     0.09
1485   0.00   -0.01      0.04       0.00    -0.01    -0.01    -0.01
1491   0.02    0.01     -0.07       0.00     0.00     0.00     0.00
```

```
1636  0.04   0.06    -0.05        0.05    -0.03     0.01    -0.04
1640  0.12  -0.05    -0.07       -0.05     0.08     0.18     0.22
1653 -0.06   0.10     0.05       -0.05     0.10    -0.05    -0.01
1731 -0.06   0.01     0.01        0.06     0.01     0.01     0.09
1918  0.38  -0.22    -0.24        0.02    -0.13    -0.13    -0.16
2015  0.02  -0.01     0.00       -0.01    -0.01     0.01    -0.01
2157  0.01   0.00    -0.01       -0.02     0.01     0.01     0.02
2167  0.04  -0.05     0.05        0.13    -0.03     0.04    -0.21
2168  0.01   0.00    -0.04       -0.03     0.02     0.03     0.04
2169 -0.03   0.23     0.06       -0.12     0.33     0.56    -1.05_*
2249 -0.04   0.02     0.05        0.02    -0.01     0.03     0.08
2272 -0.01  -0.04     0.01        0.16    -0.04    -0.07     0.11
2281 -0.09   0.13     0.14        0.23    -0.22     0.11    -0.12
     dfb.‘HMLDs‘ dfb.Drtn dfb.Impc dfb.‘HME dfb.‘SI‘ dffit   cov.r   cook.d
7     0.02       -0.01    -0.15     0.03    -0.14    -0.26    1.13_*  0.01
8    -0.08       -0.01     0.01     0.09     0.01    -0.14    1.15_*  0.00
30    0.16        0.08    -0.04    -0.10     0.00     0.35    0.82_*  0.01
178   0.00       -0.19    -0.03     0.04     0.10     0.50    0.71_*  0.02
194  -0.02       -0.02    -0.01     0.02    -0.02     0.11    1.20_*  0.00
197   0.05       -0.12     0.06    -0.02     0.05     0.32    0.86_*  0.01
236  -0.05        0.18    -0.04     0.12     0.04     0.33    0.87_*  0.01
252  -0.02       -0.03     0.03     0.06     0.06    -0.18    1.12_*  0.00
369   0.08       -0.12    -0.28    -0.06    -0.12     0.47    0.74_*  0.02
512   0.03        0.40    -0.03    -0.04    -0.22     0.58    1.12_*  0.03
643   0.02       -0.02     0.00    -0.01     0.02    -0.11    1.13_*  0.00
696  -0.05        0.12     0.07     0.00    -0.02    -0.27    0.87_*  0.01
908   0.13        0.00     0.02    -0.07    -0.15    -0.23    1.28_*  0.00
1013 -0.06       -0.05     0.02    -0.08     0.08     0.34    1.16_*  0.01
1256  0.10        0.02     0.05     0.02    -0.17    -0.32    1.36_*  0.01
1334 -0.08        0.14     0.86     0.03     0.30    -1.24_*  1.22_*  0.13
1481  0.89       -0.15     0.18    -0.95    -0.06    -2.02_*  1.29_*  0.34
1485 -0.02       -0.01     0.01     0.02     0.01     0.05    1.15_*  0.00
1491  0.04        0.00     0.00    -0.04    -0.01    -0.07    1.37_*  0.00
1636  0.03        0.03     0.03    -0.07     0.03    -0.22    1.21_*  0.00
1640  0.10       -0.36    -0.21    -0.04    -0.21    -0.53    0.88_*  0.02
1653 -0.11        0.02     0.04     0.03     0.01    -0.26    1.20_*  0.01
1731  0.04        0.03    -0.01    -0.07    -0.08     0.16    1.17_*  0.00
1918  0.11       -0.03     0.10    -0.09     0.16     0.51    0.85_*  0.02
2015  0.01        0.01     0.00     0.03     0.00    -0.09    1.12_*  0.00
2157  0.05        0.00    -0.01    -0.04    -0.02     0.08    1.18_*  0.00
2167 -0.50       -0.02    -0.02     0.41     0.21    -0.84_*  1.30_*  0.06
2168  0.12        0.00    -0.04    -0.10    -0.05     0.19    1.27_*  0.00
2169 -0.79        0.20    -0.51     0.45     1.02_*   1.55_*  0.95    0.20
2249 -0.01        0.00    -0.03     0.03    -0.08    -0.14    1.13_*  0.00
2272  0.11        0.02     0.02    -0.18    -0.10     0.37    0.83_*  0.01
2281 -0.21        0.15    -0.11     0.08     0.10    -0.50    0.82_*  0.02
     hat
7     0.10
8     0.10
30    0.02
178   0.02
194   0.14_*
197   0.02
236   0.02
252   0.08
369   0.02
512   0.14_*
643   0.08
696   0.01
908   0.20_*
```

```
1013  0.13_*
1256  0.25_*
1334  0.27_*
1481  0.38_*
1485  0.10
1491  0.24_*
1636  0.15_*
1640  0.05
1653  0.15_*
1731  0.12_*
1918  0.04
2015  0.08
2157  0.12_*
2167  0.26_*
2168  0.19_*
2169  0.22_*
2249  0.09
2272  0.02
2281  0.03
```

```
outlierTest(glmTest)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
178 3.243576          0.0011804      0.37537
```

```
########Poisson Regression
summary(poissonTest <- glm(RPE ~ `Average Metabolic Power` + Decelerations + Accelerations +
                           `Dynamic Stress Load` + Distance + `HML Distance` + Duration +
                           `HML Efforts` + `Speed Intensity` , family = "poisson", data = res$test))
```

```
Call:
glm(formula = RPE ~ `Average Metabolic Power` + Decelerations +
    Accelerations + `Dynamic Stress Load` + Distance + `HML Distance` +
    Duration + `HML Efforts` + `Speed Intensity`, family = "poisson",
    data = res$test)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.40483  -0.41907  -0.03473   0.39652   1.72605

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)                1.3103783  0.1410396   9.291  < 2e-16 ***
`Average Metabolic Power` -0.0236865  0.0394557  -0.600  0.54829
Decelerations              0.0012508  0.0023349   0.536  0.59215
Accelerations              0.0044112  0.0022255   1.982  0.04746 *
`Dynamic Stress Load`      0.0006959  0.0002787   2.497  0.01253 *
Distance                   0.0004354  0.0002226   1.956  0.05049 .
`HML Distance`             0.0004156  0.0001337   3.109  0.00188 **
Duration                  -0.0035682  0.0011834  -3.015  0.00257 **
`HML Efforts`             -0.0026435  0.0010586  -2.497  0.01252 *
`Speed Intensity`         -0.0078984  0.0046325  -1.705  0.08819 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)
```

```
    Null deviance: 237.66  on 317  degrees of freedom
Residual deviance: 137.00  on 308  degrees of freedom
AIC: Inf

Number of Fisher Scoring iterations: 4
```

```
RMSE(poissonTest) #0.6563605
```

```
[1] 0.6563605
```

```
count4 <- summary(influence.measures(poissonTest)) #40
```

```
Potentially influential observations of
     glm(formula = RPE ~ `Average Metabolic Power` + Decelerations +      Accelerations + `Dynamic Stress Load

     dfb.1_ dfb.`AMP dfb.Dclr dfb.Accl dfb.`DSL dfb.Dstn dfb.`HMD dfb.Drtn
7      0.00    0.01     0.00    -0.05    -0.04     0.10     0.05    -0.03
8      0.01    0.00    -0.04     0.01     0.05    -0.01    -0.08    -0.02
30     0.04   -0.01     0.02    -0.12    -0.09    -0.04     0.08     0.10
178    0.30   -0.23     0.00     0.10    -0.01    -0.09    -0.01    -0.16
194    0.04   -0.06    -0.12     0.16     0.02     0.03     0.00    -0.05
236   -0.01   -0.02    -0.07    -0.01    -0.03    -0.07    -0.03     0.18
252    0.05    0.00    -0.02    -0.07     0.00    -0.05    -0.03    -0.03
297    0.00    0.01    -0.05    -0.05    -0.02     0.03     0.02    -0.01
369    0.14   -0.07    -0.06    -0.09     0.03     0.04     0.02    -0.08
497   -0.31    0.19    -0.08    -0.06    -0.09    -0.02     0.02     0.49
511    0.04   -0.02     0.01    -0.01     0.00    -0.01     0.00    -0.04
512   -0.56    0.43    -0.19     0.07    -0.04     0.20     0.10     0.49
633   -0.05    0.02    -0.13     0.18     0.13     0.16     0.02    -0.33
644    0.01   -0.02    -0.01     0.00    -0.03     0.00     0.01     0.00
696   -0.20    0.15    -0.07     0.10     0.02     0.04    -0.02     0.10
718   -0.04    0.06     0.07    -0.03     0.00     0.08     0.04    -0.16
731    0.03   -0.03    -0.04     0.03    -0.04    -0.02    -0.04     0.00
747   -0.01    0.01     0.00     0.00     0.02     0.01     0.01     0.00
908   -0.01    0.04    -0.04    -0.07     0.10     0.22     0.23     0.01
1013  -0.01   -0.01     0.04    -0.01    -0.07    -0.01    -0.03    -0.03
1154   0.15   -0.26    -0.09     0.16     0.09     0.30     0.18    -0.53
1170  -0.09    0.06     0.09    -0.04     0.07    -0.03    -0.01     0.00
1256   0.01    0.02    -0.05     0.07    -0.03     0.23     0.23     0.02
1334  -0.13    0.16     0.15     0.00    -0.60    -0.10    -0.05     0.15
1382   0.03    0.05    -0.06     0.04     0.04    -0.03    -0.10     0.00
1636   0.00    0.08     0.08    -0.05     0.09    -0.06    -0.02     0.06
1640   0.13   -0.05    -0.08     0.07    -0.01     0.18     0.07    -0.39
1653  -0.08    0.17    -0.09     0.19    -0.06     0.02    -0.15     0.00
1731  -0.06    0.01     0.05     0.01     0.01     0.08     0.06     0.02
2015   0.05   -0.01    -0.02    -0.02     0.03    -0.04     0.01     0.03
2033   0.01   -0.01     0.00     0.00     0.00     0.01     0.01    -0.01
2075   0.05   -0.02     0.05    -0.02     0.02     0.10     0.05    -0.25
2157   0.00    0.00     0.00     0.00     0.00     0.00    -0.01     0.00
2167   0.12   -0.07     0.14    -0.06     0.11    -0.24    -0.70    -0.03
2168   0.00    0.00     0.01     0.00     0.00    -0.01    -0.04     0.00
2169   0.05    0.20    -0.22     0.21     0.19    -0.86    -0.80     0.14
2233   0.02   -0.04    -0.01     0.01     0.00     0.01     0.01    -0.02
2249  -0.02    0.01     0.01    -0.01     0.01     0.10     0.04    -0.02
2272   0.01   -0.04     0.14    -0.04    -0.10     0.09     0.12     0.03
2281   0.00    0.11     0.23    -0.26     0.03    -0.13    -0.13     0.12
     dfb.`HME dfb.`SI` dffit   cov.r   cook.d hat
7      0.01    -0.10    -0.18    1.11_*  0.00   0.08
8      0.10     0.00    -0.14    1.15_*  0.00   0.11_*
```

```
30   -0.03     0.04     0.28    0.89_*  0.00   0.01
178   0.05     0.10     0.39    0.84_*  0.01   0.02
194   0.00    -0.03     0.17    1.18_*  0.00   0.13_*
236   0.10     0.06     0.29    0.90_*  0.00   0.02
252   0.07     0.05    -0.15    1.12_*  0.00   0.09
297  -0.02    -0.02    -0.15    1.13_*  0.00   0.09
369   0.00    -0.03     0.27    0.85_*  0.00   0.01
497   0.03    -0.01     0.58_*  0.92    0.02   0.06
511   0.02     0.01    -0.07    1.12_*  0.00   0.08
512  -0.13    -0.23     0.67_*  1.08    0.02   0.13_*
633  -0.01    -0.14    -0.54_*  0.68_*  0.01   0.02
644  -0.02     0.00    -0.06    1.10_*  0.00   0.06
696  -0.05    -0.04    -0.25    0.90_*  0.00   0.01
718  -0.09    -0.07    -0.31    0.89_*  0.00   0.02
731   0.03     0.02    -0.10    1.12_*  0.00   0.08
747  -0.01    -0.01     0.02    1.13_*  0.00   0.08
908  -0.10    -0.24    -0.35    1.37_*  0.01   0.26_*
1013 -0.06     0.02     0.17    1.20_*  0.00   0.14_*
1154 -0.16    -0.25    -0.62_*  0.69_*  0.01   0.03
1170  0.01     0.02    -0.24    0.86_*  0.00   0.01
1256  0.01    -0.26    -0.48    1.43_*  0.01   0.29_*
1334  0.00     0.09    -0.66_*  1.13_*  0.02   0.15_*
1382  0.00     0.03    -0.29    1.10_*  0.00   0.09
1636 -0.06     0.05    -0.27    1.28_*  0.00   0.20_*
1640  0.01    -0.17    -0.48    0.86_*  0.01   0.03
1653 -0.05    -0.02    -0.46    1.26_*  0.01   0.20_*
1731 -0.09    -0.07     0.14    1.20_*  0.00   0.14_*
2015  0.08     0.03    -0.17    1.14_*  0.00   0.10_*
2033  0.00     0.00    -0.02    1.10_*  0.00   0.06
2075 -0.03    -0.09    -0.35    0.90_*  0.00   0.02
2157  0.00     0.00    -0.01    1.16_*  0.00   0.11_*
2167  0.58     0.24    -1.05_*  1.45_*  0.05   0.34_*
2168  0.03     0.01    -0.06    1.28_*  0.00   0.20_*
2169  0.50     0.86     1.12_*  1.05    0.07   0.18_*
2233 -0.01     0.00    -0.05    1.10_*  0.00   0.06
2249 -0.01    -0.10    -0.14    1.11_*  0.00   0.08
2272 -0.21    -0.08     0.30    0.90_*  0.00   0.02
2281 -0.04     0.12    -0.47    0.90_*  0.01   0.04
```

```
outlierTest(poissonTest)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
     rstudent unadjusted p-value Bonferroni p
633 -2.417635           0.015622           NA
```

```
########Penalized Regression

#################Ridge
xTest <- model.matrix(RPE~., res$test)[,-1]
yTest <- res$test$RPE
glmnet(xTest, yTest, alpha = 1, lambda = NULL)
```

```
Call:  glmnet(x = xTest, y = yTest, alpha = 1, lambda = NULL)

   Df  %Dev  Lambda
1   0  0.00 1.11000
2   1  6.20 1.01200
3   1 11.35 0.92190
```

```
4   1 15.63 0.84000
5   1 19.18 0.76530
6   1 22.12 0.69740
7   1 24.57 0.63540
8   3 26.92 0.57900
9   3 29.18 0.52750
10  4 31.18 0.48070
11  5 33.37 0.43800
12  5 35.81 0.39910
13  5 37.84 0.36360
14  7 39.60 0.33130
15  7 41.18 0.30190
16  7 42.49 0.27510
17  7 43.59 0.25060
18  7 44.49 0.22840
19  7 45.25 0.20810
20  7 45.87 0.18960
21  8 46.39 0.17270
22  7 46.86 0.15740
23  7 47.23 0.14340
24  7 47.53 0.13070
25  8 48.05 0.11910
26  8 48.51 0.10850
27  8 48.89 0.09885
28  8 49.20 0.09007
29  9 49.72 0.08207
30  9 50.25 0.07478
31  9 50.69 0.06813
32  9 51.05 0.06208
33  9 51.35 0.05656
34 10 51.61 0.05154
35 10 51.89 0.04696
36 10 52.12 0.04279
37 11 52.31 0.03899
38 11 52.47 0.03552
39 11 52.61 0.03237
40 11 52.72 0.02949
41 11 52.82 0.02687
42 11 52.89 0.02449
43 12 52.99 0.02231
44 12 53.07 0.02033
45 12 53.14 0.01852
46 12 53.20 0.01688
47 12 53.24 0.01538
48 12 53.28 0.01401
49 12 53.32 0.01277
50 12 53.34 0.01163
51 12 53.36 0.01060
52 12 53.38 0.00966
53 12 53.40 0.00880
54 12 53.41 0.00802
55 12 53.42 0.00731
56 12 53.43 0.00666
57 12 53.44 0.00606
58 14 53.45 0.00553
59 14 53.47 0.00504
60 14 53.49 0.00459
61 14 53.51 0.00418
62 14 53.52 0.00381
63 14 53.53 0.00347
```

```
64 14 53.54 0.00316
65 14 53.55 0.00288
66 14 53.56 0.00263
67 14 53.56 0.00239
68 14 53.57 0.00218
69 14 53.57 0.00199
70 14 53.57 0.00181
71 14 53.58 0.00165
72 14 53.58 0.00150
73 14 53.58 0.00137
74 14 53.58 0.00125
75 14 53.58 0.00114
76 14 53.58 0.00104
77 14 53.59 0.00094
78 14 53.59 0.00086
79 14 53.59 0.00078
80 14 53.59 0.00071
```

```
cvTest <- cv.glmnet(xTest, yTest, alpha = 0)
# Display the best lambda value
cvTest$lambda.min
```

```
[1] 0.1610982
```

```
# Fit the final model on the test data
ridgeTest <- glmnet(xTest, yTest, alpha = 0, lambda = cvTest$lambda.min)
# Display regression coefficients
coef(ridgeTest)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                                      s0
(Intercept)                   2.4744209878
Duration                     -0.0072816014
Distance                      0.0001090408
`Max Speed`                  -0.0230689452
`HML Distance`                0.0004446671
`HML Efforts`                 0.0005744979
`Sprint Distance`            -0.0005976758
Sprints                      -0.0433652488
Accelerations                 0.0164402783
Decelerations                 0.0100168213
`Average Metabolic Power`     0.0456111256
`Dynamic Stress Load`         0.0023770291
`High Speed Running (Relative)` 0.0020690653
`HML Density`                 0.1020118771
`Speed Intensity`             0.0021226404
Impacts                       0.0030072322
```

```
lambda <- 10^seq(-3, 3, length = 100)
# Build the model
set.seed(42)
ridgeTest1 <- train(
  RPE ~., data = res$test, method = "glmnet",
  trControl = trainControl("cv", number = 10),
  tuneGrid = expand.grid(alpha = 0, lambda = lambda)
)


RMSE(ridgeTest1) #1.262414
```

```
[1] 1.262414
```

```
#count5 <- summary(influence.measures(ridgeTest1$finalModel)) #51

#outlierTest(ridgeTest1)

################Lasso
xTestLasso <- model.matrix(RPE ~ `Average Metabolic Power` + `HML Efforts` + Decelerations + Accelerations +
                             `Dynamic Stress Load` + Distance, res$test)[,-1]
yTestLasso <- res$test$RPE
glmnet(xTestLasso, yTestLasso, alpha = 1, lambda = NULL)
```

```
Call:  glmnet(x = xTestLasso, y = yTestLasso, alpha = 1, lambda = NULL)

   Df  %Dev  Lambda
1   0  0.00 0.98950
2   1  4.93 0.90160
3   3  9.87 0.82150
4   3 14.57 0.74850
5   3 18.48 0.68200
6   3 21.73 0.62140
7   3 24.42 0.56620
8   4 26.92 0.51590
9   4 29.07 0.47010
10  4 30.86 0.42830
11  4 32.35 0.39030
12  4 33.58 0.35560
13  4 34.60 0.32400
14  4 35.45 0.29520
15  4 36.16 0.26900
16  4 36.75 0.24510
17  4 37.23 0.22330
18  4 37.64 0.20350
19  4 37.97 0.18540
20  5 38.25 0.16890
21  5 38.48 0.15390
22  5 38.68 0.14030
23  5 38.84 0.12780
24  5 38.98 0.11640
25  5 39.09 0.10610
26  5 39.18 0.09667
27  5 39.26 0.08809
28  5 39.32 0.08026
29  5 39.37 0.07313
30  5 39.42 0.06663
31  5 39.46 0.06071
32  5 39.49 0.05532
33  5 39.51 0.05041
34  5 39.53 0.04593
35  5 39.55 0.04185
36  5 39.56 0.03813
37  5 39.58 0.03474
38  5 39.59 0.03166
39  5 39.59 0.02884
40  5 39.60 0.02628
41  5 39.61 0.02395
42  5 39.61 0.02182
43  5 39.62 0.01988
44  5 39.62 0.01811
```

```
45  6 39.65 0.01651
46  6 39.68 0.01504
47  6 39.71 0.01370
48  6 39.73 0.01249
49  6 39.74 0.01138
50  6 39.76 0.01037
51  6 39.77 0.00944
52  6 39.78 0.00861
53  6 39.79 0.00784
54  6 39.79 0.00715
55  6 39.80 0.00651
56  6 39.80 0.00593
57  6 39.81 0.00540
58  6 39.81 0.00492
59  6 39.81 0.00449
60  6 39.82 0.00409
61  6 39.82 0.00372
62  6 39.82 0.00339
63  6 39.82 0.00309
64  6 39.82 0.00282
65  6 39.82 0.00257
66  6 39.82 0.00234
67  6 39.82 0.00213
68  6 39.82 0.00194
69  6 39.83 0.00177
70  6 39.83 0.00161
```

```
cvTest <- cv.glmnet(xTestLasso, yTestLasso, alpha = 1)
# Display the best lambda value
cvTest$lambda.min
```

```
[1] 0.001612612
```

```
# Fit the final model on the test data
lassoTest <- glmnet(xTestLasso, yTestLasso, alpha = 1, lambda = cvTest$lambda.min)
# Dsiplay regression coefficients
coef(lassoTest)
```

```
7 x 1 sparse Matrix of class "dgCMatrix"
                                    s0
(Intercept)                1.8905610821
‘Average Metabolic Power‘  0.3640157724
‘HML Efforts‘             -0.0024756223
Decelerations              0.0045891748
Accelerations              0.0245743801
‘Dynamic Stress Load‘      0.0029162621
Distance                   0.0001005957
```

```
lassoTest1 <- train(
  RPE ~ `Average Metabolic Power` + `HML Efforts` + Decelerations + Accelerations +
    `Dynamic Stress Load` + Distance, data = res$test, method = "glmnet",
  trControl = trainControl("cv", number = 10),
  tuneGrid = expand.grid(alpha = 0, lambda = lambda)
)
```

```
lassoTest1
```

```
glmnet
```

```
318 samples
  6 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 286, 286, 286, 286, 288, 285, ...
Resampling results across tuning parameters:
```

lambda	RMSE	Rsquared	MAE
1.000000e-03	1.437078	0.3974461	1.163088
1.149757e-03	1.437078	0.3974461	1.163088
1.321941e-03	1.437078	0.3974461	1.163088
1.519911e-03	1.437078	0.3974461	1.163088
1.747528e-03	1.437078	0.3974461	1.163088
2.009233e-03	1.437078	0.3974461	1.163088
2.310130e-03	1.437078	0.3974461	1.163088
2.656088e-03	1.437078	0.3974461	1.163088
3.053856e-03	1.437078	0.3974461	1.163088
3.511192e-03	1.437078	0.3974461	1.163088
4.037017e-03	1.437078	0.3974461	1.163088
4.641589e-03	1.437078	0.3974461	1.163088
5.336699e-03	1.437078	0.3974461	1.163088
6.135907e-03	1.437078	0.3974461	1.163088
7.054802e-03	1.437078	0.3974461	1.163088
8.111308e-03	1.437078	0.3974461	1.163088
9.326033e-03	1.437078	0.3974461	1.163088
1.072267e-02	1.437078	0.3974461	1.163088
1.232847e-02	1.437078	0.3974461	1.163088
1.417474e-02	1.437078	0.3974461	1.163088
1.629751e-02	1.437078	0.3974461	1.163088
1.873817e-02	1.437078	0.3974461	1.163088
2.154435e-02	1.437078	0.3974461	1.163088
2.477076e-02	1.437078	0.3974461	1.163088
2.848036e-02	1.437078	0.3974461	1.163088
3.274549e-02	1.437078	0.3974461	1.163088
3.764936e-02	1.437078	0.3974461	1.163088
4.328761e-02	1.437078	0.3974461	1.163088
4.977024e-02	1.437078	0.3974461	1.163088
5.722368e-02	1.437078	0.3974461	1.163088
6.579332e-02	1.437078	0.3974461	1.163088
7.564633e-02	1.437078	0.3974461	1.163088
8.697490e-02	1.437078	0.3974461	1.163088
1.000000e-01	1.437119	0.3974690	1.163051
1.149757e-01	1.436980	0.3977131	1.162592
1.321941e-01	1.436787	0.3979729	1.162045
1.519911e-01	1.436635	0.3982288	1.161457
1.747528e-01	1.436544	0.3984640	1.160837
2.009233e-01	1.436532	0.3986706	1.160185
2.310130e-01	1.436617	0.3988401	1.159530
2.656088e-01	1.436812	0.3989688	1.158859
3.053856e-01	1.437138	0.3990457	1.158185
3.511192e-01	1.437618	0.3990624	1.157570
4.037017e-01	1.438272	0.3990105	1.156991
4.641589e-01	1.439121	0.3988890	1.156502
5.336699e-01	1.440190	0.3986898	1.156143
6.135907e-01	1.441498	0.3984193	1.155956
7.054802e-01	1.443071	0.3980775	1.155895
8.111308e-01	1.444948	0.3976664	1.155899
9.326033e-01	1.447157	0.3971968	1.156014

```
  1.072267e+00  1.449746  0.3966748  1.156592
  1.232847e+00  1.452762  0.3961144  1.157766
  1.417474e+00  1.456276  0.3955212  1.159240
  1.629751e+00  1.460338  0.3949135  1.161153
  1.873817e+00  1.465038  0.3942975  1.164317
  2.154435e+00  1.470448  0.3936876  1.168197
  2.477076e+00  1.476669  0.3930883  1.172732
  2.848036e+00  1.483765  0.3925119  1.178258
  3.274549e+00  1.491834  0.3919599  1.184948
  3.764936e+00  1.500911  0.3914421  1.192970
  4.328761e+00  1.511062  0.3909573  1.202343
  4.977024e+00  1.522245  0.3905073  1.213341
  5.722368e+00  1.534481  0.3900915  1.225008
  6.579332e+00  1.547652  0.3897134  1.237247
  7.564633e+00  1.561707  0.3893683  1.250391
  8.697490e+00  1.576440  0.3890581  1.263476
  1.000000e+01  1.591749  0.3887777  1.276416
  1.149757e+01  1.607367  0.3885279  1.289042
  1.321941e+01  1.623172  0.3883038  1.301303
  1.519911e+01  1.638879  0.3881056  1.313049
  1.747528e+01  1.654382  0.3879288  1.324260
  2.009233e+01  1.669422  0.3877732  1.334817
  2.310130e+01  1.683936  0.3876350  1.344736
  2.656088e+01  1.697719  0.3875140  1.353936
  3.053856e+01  1.710761  0.3874070  1.362460
  3.511192e+01  1.722921  0.3873134  1.370262
  4.037017e+01  1.734238  0.3872308  1.377405
  4.641589e+01  1.744628  0.3871588  1.383878
  5.336699e+01  1.754167  0.3870954  1.389754
  6.135907e+01  1.762816  0.3870403  1.395022
  7.054802e+01  1.770668  0.3869918  1.399760
  8.111308e+01  1.777716  0.3869497  1.403987
  9.326033e+01  1.784058  0.3869127  1.407777
  1.072267e+02  1.789705  0.3868807  1.411144
  1.232847e+02  1.794752  0.3868525  1.414137
  1.417474e+02  1.799217  0.3868281  1.416801
  1.629751e+02  1.803185  0.3868067  1.419173
  1.873817e+02  1.806680  0.3867882  1.421295
  2.154435e+02  1.809772  0.3867719  1.423219
  2.477076e+02  1.812485  0.3867579  1.424921
  2.848036e+02  1.814878  0.3867455  1.426440
  3.274549e+02  1.816972  0.3867349  1.427769
  3.764936e+02  1.818813  0.3867255  1.428937
  4.328761e+02  1.820421  0.3867175  1.429956
  4.977024e+02  1.821832  0.3867104  1.430866
  5.722368e+02  1.823062  0.3867043  1.431672
  6.579332e+02  1.824141  0.3866989  1.432429
  7.564633e+02  1.825079  0.3866943  1.433106
  8.697490e+02  1.826033  0.3866903  1.433783
  1.000000e+03  1.831037  0.2394926  1.437453

Tuning parameter 'alpha' was held constant at a value of 0
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were alpha = 0 and lambda = 0.2009233.

RMSE(lassoTest1) #1.428043

[1] 1.428043

#count6 <- summary(influence.measures(lassoTest1)) #42
```

```r
#outlierTest(lassoTest1)

################Elastic Net
# Build the model using the test set
set.seed(42)
elasticTest <- train(
  RPE ~ `Average Metabolic Power` + `HML Efforts` + Decelerations + Accelerations +
    `Dynamic Stress Load` + `Speed Intensity` + Distance + `HML Distance` +
    Duration + `Sprint Distance`, data = res$test, method = "glmnet",
  trControl = trainControl("cv", number = 5),
  tuneLength = 10
)
# Best tuning parameter
elasticTest$bestTune
```

```
   alpha     lambda
15   0.2 0.01461122
```

```r
elasticTest
```

```
glmnet

318 samples
 10 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 255, 255, 254, 254, 254
Resampling results across tuning parameters:

  alpha  lambda        RMSE      Rsquared   MAE
  0.1    0.0005130279  1.366200  0.4652068  1.089573
  0.1    0.0011851610  1.365982  0.4653106  1.089742
  0.1    0.0027378756  1.365089  0.4645379  1.091267
  0.1    0.0063248478  1.364739  0.4629243  1.091854
  0.1    0.0146112187  1.363782  0.4613833  1.091204
  0.1    0.0337538104  1.365192  0.4575624  1.093105
  0.1    0.0779756798  1.369607  0.4517917  1.097577
  0.1    0.1801339338  1.382858  0.4399513  1.106423
  0.1    0.4161327504  1.394674  0.4318627  1.115785
  0.1    0.9613206259  1.421687  0.4218790  1.132509
  0.2    0.0005130279  1.368150  0.4643652  1.089206
  0.2    0.0011851610  1.366251  0.4650924  1.089781
  0.2    0.0027378756  1.365409  0.4641920  1.091488
  0.2    0.0063248478  1.364686  0.4627281  1.091890
  0.2    0.0146112187  1.363781  0.4610336  1.091297
  0.2    0.0337538104  1.365817  0.4566471  1.093912
  0.2    0.0779756798  1.372208  0.4499843  1.099575
  0.2    0.1801339338  1.390036  0.4349417  1.111781
  0.2    0.4161327504  1.404559  0.4277962  1.122134
  0.2    0.9613206259  1.455141  0.4083184  1.151275
  0.3    0.0005130279  1.368474  0.4641738  1.089345
  0.3    0.0011851610  1.366551  0.4648480  1.089839
  0.3    0.0027378756  1.365790  0.4637887  1.091754
  0.3    0.0063248478  1.364621  0.4625012  1.091963
  0.3    0.0146112187  1.363921  0.4606087  1.091478
  0.3    0.0337538104  1.366026  0.4563848  1.094240
  0.3    0.0779756798  1.376668  0.4467640  1.102538
  0.3    0.1801339338  1.393250  0.4335936  1.114239
```

0.3	0.4161327504	1.418523	0.4211406	1.130500
0.3	0.9613206259	1.482917	0.4044879	1.175443
0.4	0.0005130279	1.368808	0.4639238	1.089474
0.4	0.0011851610	1.366903	0.4645792	1.089877
0.4	0.0027378756	1.366254	0.4633329	1.092040
0.4	0.0063248478	1.364402	0.4623877	1.091965
0.4	0.0146112187	1.364127	0.4600973	1.091623
0.4	0.0337538104	1.366167	0.4562351	1.094367
0.4	0.0779756798	1.382218	0.4426170	1.106115
0.4	0.1801339338	1.397422	0.4317237	1.117479
0.4	0.4161327504	1.435116	0.4127206	1.139743
0.4	0.9613206259	1.515445	0.4019363	1.209540
0.5	0.0005130279	1.369173	0.4636197	1.089650
0.5	0.0011851610	1.367298	0.4642518	1.089956
0.5	0.0027378756	1.366777	0.4627939	1.092368
0.5	0.0063248478	1.364276	0.4621561	1.092090
0.5	0.0146112187	1.364588	0.4593591	1.092189
0.5	0.0337538104	1.366730	0.4557911	1.094892
0.5	0.0779756798	1.388492	0.4377020	1.110660
0.5	0.1801339338	1.402656	0.4291975	1.121371
0.5	0.4161327504	1.448395	0.4077758	1.147277
0.5	0.9613206259	1.551635	0.3967197	1.245189
0.6	0.0005130279	1.369556	0.4633013	1.089817
0.6	0.0011851610	1.367767	0.4638713	1.090026
0.6	0.0027378756	1.367119	0.4624032	1.092470
0.6	0.0063248478	1.364293	0.4619363	1.092162
0.6	0.0146112187	1.365110	0.4585656	1.092862
0.6	0.0337538104	1.367706	0.4551029	1.095667
0.6	0.0779756798	1.391873	0.4352989	1.113025
0.6	0.1801339338	1.408370	0.4264380	1.125280
0.6	0.4161327504	1.457706	0.4084976	1.156197
0.6	0.9613206259	1.592177	0.3881010	1.280300
0.7	0.0005130279	1.369918	0.4630209	1.089925
0.7	0.0011851610	1.368335	0.4634296	1.090122
0.7	0.0027378756	1.367295	0.4621935	1.092541
0.7	0.0063248478	1.364325	0.4617165	1.092209
0.7	0.0146112187	1.365674	0.4577218	1.093516
0.7	0.0337538104	1.369470	0.4538261	1.096743
0.7	0.0779756798	1.393310	0.4347140	1.113806
0.7	0.1801339338	1.414928	0.4229883	1.128725
0.7	0.4161327504	1.468250	0.4075609	1.165675
0.7	0.9613206259	1.632541	0.3799502	1.310632
0.8	0.0005130279	1.370318	0.4626969	1.090049
0.8	0.0011851610	1.368987	0.4628995	1.090204
0.8	0.0027378756	1.367397	0.4620081	1.092531
0.8	0.0063248478	1.364371	0.4615062	1.092243
0.8	0.0146112187	1.366035	0.4571358	1.093942
0.8	0.0337538104	1.371782	0.4520948	1.098230
0.8	0.0779756798	1.394656	0.4342142	1.114486
0.8	0.1801339338	1.421579	0.4197053	1.131767
0.8	0.4161327504	1.480522	0.4052429	1.176370
0.8	0.9613206259	1.669926	0.3745579	1.337352
0.9	0.0005130279	1.370815	0.4623114	1.090224
0.9	0.0011851610	1.369813	0.4622627	1.090644
0.9	0.0027378756	1.367564	0.4617699	1.092539
0.9	0.0063248478	1.364434	0.4612760	1.092289
0.9	0.0146112187	1.366095	0.4569129	1.093997
0.9	0.0337538104	1.374411	0.4500971	1.100208
0.9	0.0779756798	1.396003	0.4337590	1.115606
0.9	0.1801339338	1.427781	0.4170783	1.135067

```
  0.9    0.4161327504  1.494694  0.4012543  1.189006
  0.9    0.9613206259  1.707162  0.3742786  1.363276
  1.0    0.0005130279  1.371381  0.4618498  1.090432
  1.0    0.0011851610  1.370810  0.4614542  1.091157
  1.0    0.0027378756  1.367689  0.4616067  1.092541
  1.0    0.0063248478  1.364515  0.4610337  1.092341
  1.0    0.0146112187  1.366104  0.4567948  1.093918
  1.0    0.0337538104  1.377361  0.4478296  1.102325
  1.0    0.0779756798  1.397549  0.4331883  1.116786
  1.0    0.1801339338  1.433412  0.4149858  1.138827
  1.0    0.4161327504  1.509947  0.3959194  1.202854
  1.0    0.9613206259  1.751104  0.3742786  1.392223

RMSE was used to select the optimal model using the smallest value.
The final values used for the model were alpha = 0.2 and lambda = 0.01461122.

# Coefficient of the final model. You need to specify the best lambda
coef(elasticTest$finalModel, elasticTest$bestTune$lambda)

11 x 1 sparse Matrix of class "dgCMatrix"
                                    s1
(Intercept)                3.535448e+00
‘Average Metabolic Power‘ -9.165482e-02
‘HML Efforts‘             -1.201850e-02
Decelerations              9.761720e-03
Accelerations              1.769520e-02
‘Dynamic Stress Load‘      2.967801e-03
‘Speed Intensity‘          9.170056e-06
Distance                   2.815872e-04
‘HML Distance‘             1.859634e-03
Duration                  -1.297910e-02
‘Sprint Distance‘         -7.593371e-04

RMSE(elasticTest) #1.319529

[1] 1.319529

#count7 <- summary(influence.measures(elasticTest)) #37
#outlierTest(elasticTest)

########SVM
OptModelsvmTest <- tune(svm, RPE ~ `Average Metabolic Power` + Distance +
                          `Speed Intensity` + `HML Distance` + `Dynamic Stress Load` +
                          `HML Efforts` + Decelerations + Sprints + Accelerations +
                          Impacts, data=res$test,ranges=list(elsilon=seq(0,1,0.1), cost=1:100))
OptModelsvmTest

Parameter tuning of ’svm’:

- sampling method: 10-fold cross validation

- best parameters:
 elsilon cost
       0    1

- best performance: 1.726158

bestSVMTest <- OptModelsvmTest$best.model
bestSVMTest
```

```
Call:
best.tune(method = svm, train.x = RPE ~ `Average Metabolic Power` +
    Distance + `Speed Intensity` + `HML Distance` + `Dynamic Stress Load` +
    `HML Efforts` + Decelerations + Sprints + Accelerations + Impacts,
    data = res$test, ranges = list(elsilon = seq(0, 1, 0.1), cost = 1:100))


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.1
    epsilon:  0.1


Number of Support Vectors:  276

RMSE(bestSVMTest) #1.128293

[1] 1.128293

#count8 <- summary(influence.measures(svm.model)) #33
#outlierTest(svm.model)

########Boosted Forests
set.seed(42)
boostedTest <- train(
  RPE ~ `HML Distance` + `Average Metabolic Power` + Accelerations + Duration + `Speed Intensity` +
    Decelerations + `Sprint Distance` + `Dynamic Stress Load`, data = res$test, method = "xgbTree",
  trControl = trainControl("cv", number = 10), verbosity = 0
)
RMSE(boostedTest) #0.8159172

[1] 0.8159172

#count9 <- summary(influence.measures(boostedTest)) #36
#outlierTest(boostedTest)

######################################## SVM has Best Model
OptModelsvmValidate <- tune(svm, RPE ~ `Average Metabolic Power` + Distance +
                              `Speed Intensity` + `HML Distance` + `Dynamic Stress Load` +
                              `HML Efforts` + Decelerations + Sprints + Accelerations +
                              Impacts, data=res$validate,ranges=list(elsilon=seq(0,1,0.1), cost=1:100))
OptModelsvmValidate

Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 elsilon cost
       0    2

- best performance: 1.897129

bestSVMTValidate <- OptModelsvmValidate$best.model
bestSVMTValidate
```

```
Call:
best.tune(method = svm, train.x = RPE ~ `Average Metabolic Power` +
    Distance + `Speed Intensity` + `HML Distance` + `Dynamic Stress Load` +
    `HML Efforts` + Decelerations + Sprints + Accelerations + Impacts,
    data = res$validate, ranges = list(elsilon = seq(0, 1, 0.1),
        cost = 1:100))


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  2
      gamma:  0.1
    epsilon:  0.1


Number of Support Vectors:  282

RMSE(bestSVMTValidate) #1.128293 to 1.074248

[1] 1.11796

#count11 <- summary(influence.measures(bestSVMTValidate)) #32
#outlierTest(bestSVMTValidate)

########KNN
train.loanVal <- res$validate[c(1,11,3,15,5,12,6,10,8,9,16)] # 70% training data
test.loanVal <- res$test[c(1,11,3,15,5,12,6,10,8,9,16)] # remaining 30% test data
#Creating seperate dataframe for rpe feature which is the target.
train.loan_labelsVal <- res$validate[,1]
head(train.loan_labelsVal)

[1]  6.0 10.0 10.0  7.0  7.0  7.5

test.loan_labelsVal <-res$test[,1]
head(test.loan_labelsVal)

[1] 8 7 6 7 5 8

#Find the number of observation
NROW(train.loan_labelsVal) #17.83, rows 318

[1] 318

set.seed(42)
knnVal.17 <- knn(train=train.loanVal, test=test.loanVal, cl=train.loan_labelsVal, k=17)
set.seed(42)
knnVal.18 <- knn(train=train.loanVal, test=test.loanVal, cl=train.loan_labelsVal, k=18)
#Calculate the proportion of correct classification for k = 17, 18
set.seed(42)
ACCval.17 <- 100 * sum(test.loan_labelsVal == knnVal.17)/NROW(test.loan_labelsVal)
set.seed(42)
ACCval.18 <- 100 * sum(test.loan_labelsVal == knnVal.18)/NROW(test.loan_labelsVal)
ACCval.17

[1] 27.35849
```

```
ACCval.18
```

```
[1] 26.10063
```

```
# Check prediction against actual value in tabular form for k=17
table(knnVal.17 ,test.loan_labelsVal)
```

```
         test.loan_labelsVal
knnVal.17  0  1  2  3 3.5  4 4.5  5  6 6.5  7  8 8.5  9
      1    0  0  1  1   0  2   0  0  0   0  0  0   0  0
      2    0  3  8  4   1  5   0  1  2   0  0  0   0  0
      3    2  3  7 11   0  8   0  5  1   0  4  1   0  1
      3.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
      4    0  1  1  5   0 28   0 18  8   0  6  3   0  0
      5    0  1  2  6   0  9   0 18  9   0  9  5   0  1
      5.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
      6    0  0  0  2   0 14   0 27 11   1 21  4   0  4
      7    0  0  0  1   0  2   0  4  4   0  7  2   0  0
      7.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
      8    0  0  0  0   0  1   1  1  0   0  4  4   2  0
      9    0  0  0  0   0  0   0  0  0   0  0  0   0  0
      10   0  0  0  0   0  0   0  0  0   0  0  0   0  0
```

```
cor(as.numeric(knnVal.17), test.loan_labelsVal)
```

```
[1] 0.5297282
```

```
# Check prediction against actual value in tabular form for k=18
table(knnVal.18 ,test.loan_labelsVal)
```

```
         test.loan_labelsVal
knnVal.18  0  1  2  3 3.5  4 4.5  5  6 6.5  7  8 8.5  9
      1    0  0  1  1   0  2   0  0  0   0  1  0   0  0
      2    2  1  6  3   1  2   0  2  2   0  0  0   0  0
      3    0  5  9 10   0 12   0  3  0   0  3  1   0  1
      3.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
      4    0  0  1  5   0 29   1 22  8   0  5  2   0  1
      5    0  2  2  7   0 10   0 13  9   0 10  7   0  1
      5.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
      6    0  0  0  3   0 13   0 28 12   1 21  4   0  3
      7    0  0  0  1   0  1   0  5  4   0  9  1   0  0
      7.5  0  0  0  0   0  0   0  0  0   0  0  0   0  0
      8    0  0  0  0   0  0   0  1  0   0  2  4   2  0
      9    0  0  0  0   0  0   0  0  0   0  0  0   0  0
      10   0  0  0  0   0  0   0  0  0   0  0  0   0  0
```

```
cor(as.numeric(knnVal.18), test.loan_labelsVal)
```

```
[1] 0.5238586
```

```
uKNNval <- union(knnVal.18, res$validate$RPE)
tKNNval <- table(factor(knnVal.18, uKNNval), factor(res$validate$RPE, uKNNval))
confusionMatrix(tKNNval)
```

```
Confusion Matrix and Statistics
```

```
      4  6  5  7  3  2  8  1 10 7.5  9 5.5 3.5
  4   17 13 14  6 11  1  7  2  0   1  1   0   1
  6   17 11 15 19  3  5  6  5  2   0  1   1   0
  5    9 13 12  6 10  4  5  1  0   1  0   0   0
  7    4  4  7  3  2  0  0  0  0   1  0   0   0
  3    5 12 11  4  4  2  2  3  0   0  1   0   0
  2    4  3  4  1  2  0  2  2  1   0  0   0   0
  8    0  1  0  1  5  0  1  1  0   0  0   0   0
  1    0  0  1  0  3  1  0  0  0   0  0   0   0
  10   0  0  0  0  0  0  0  0  0   0  0   0   0
  7.5  0  0  0  0  0  0  0  0  0   0  0   0   0
  9    0  0  0  0  0  0  0  0  0   0  0   0   0
  5.5  0  0  0  0  0  0  0  0  0   0  0   0   0
  3.5  0  0  0  0  0  0  0  0  0   0  0   0   0

Overall Statistics

               Accuracy : 0.1509
                 95% CI : (0.1134, 0.1951)
    No Information Rate : 0.2013
    P-Value [Acc > NIR] : 0.9913

                  Kappa : -0.0088

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 4 Class: 6 Class: 5 Class: 7 Class: 3 Class: 2
Sensitivity           0.30357  0.19298  0.18750 0.075000  0.10000  0.00000
Specificity           0.78244  0.71648  0.80709 0.935252  0.85612  0.93770
Pos Pred Value        0.22973  0.12941  0.19672 0.142857  0.09091  0.00000
Neg Pred Value        0.84016  0.80258  0.79767 0.875421  0.86861  0.95652
Prevalence            0.17610  0.17925  0.20126 0.125786  0.12579  0.04088
Detection Rate        0.05346  0.03459  0.03774 0.009434  0.01258  0.00000
Detection Prevalence  0.23270  0.26730  0.19182 0.066038  0.13836  0.05975
Balanced Accuracy     0.54301  0.45473  0.49729 0.505126  0.47806  0.46885
                     Class: 8 Class: 1 Class: 10 Class: 7.5 Class: 9 Class: 5.5
Sensitivity          0.043478  0.00000  0.000000   0.000000 0.000000   0.000000
Specificity          0.972881  0.98355  1.000000   1.000000 1.000000   1.000000
Pos Pred Value       0.111111  0.00000       NaN        NaN      NaN        NaN
Neg Pred Value       0.928803  0.95527  0.990566   0.990566 0.990566   0.996855
Prevalence           0.072327  0.04403  0.009434   0.009434 0.009434   0.003145
Detection Rate       0.003145  0.00000  0.000000   0.000000 0.000000   0.000000
Detection Prevalence 0.028302  0.01572  0.000000   0.000000 0.000000   0.000000
Balanced Accuracy    0.508180  0.49178  0.500000   0.500000 0.500000   0.500000
                     Class: 3.5
Sensitivity            0.000000
Specificity            1.000000
Pos Pred Value              NaN
Neg Pred Value         0.996855
Prevalence             0.003145
Detection Rate         0.000000
Detection Prevalence   0.000000
Balanced Accuracy      0.500000

#optimizing the accuracy
i=1
k.optm=1
set.seed(42)
```

```r
for (i in 1:75){
  knn.modVal <- knn(train=train.loanVal, test=test.loanVal,cl=train.loan_labelsVal, k=i)
  k.optm[i] <- 100 * sum(test.loan_labelsVal == knn.modVal)/NROW(test.loan_labelsVal)
  k=i
  cat(k,'=',k.optm[i],'
      ')
}
```

```
1 = 19.81132
      2 = 25.4717
      3 = 18.86792
      4 = 19.18239
      5 = 21.69811
      6 = 23.89937
      7 = 21.38365
      8 = 23.58491
      9 = 23.58491
      10 = 22.64151
      11 = 24.21384
      12 = 23.27044
      13 = 26.41509
      14 = 24.21384
      15 = 25.15723
      16 = 25.78616
      17 = 28.30189
      18 = 26.72956
      19 = 25.4717
      20 = 26.72956
      21 = 29.55975
      22 = 27.04403
      23 = 29.55975
      24 = 27.67296
      25 = 27.67296
      26 = 27.04403
      27 = 26.41509
      28 = 29.24528
      29 = 26.10063
      30 = 31.44654
      31 = 27.98742
      32 = 26.41509
      33 = 27.67296
      34 = 27.04403
      35 = 25.4717
      36 = 25.78616
      37 = 25.15723
      38 = 24.5283
      39 = 24.5283
      40 = 24.84277
      41 = 24.5283
      42 = 23.89937
      43 = 23.58491
      44 = 25.4717
      45 = 24.5283
      46 = 23.27044
      47 = 21.69811
      48 = 23.27044
      49 = 22.32704
      50 = 23.58491
      51 = 22.64151
      52 = 22.64151
```

```
      53 = 23.27044
      54 = 22.32704
      55 = 21.06918
      56 = 22.95597
      57 = 22.64151
      58 = 22.32704
      59 = 21.06918
      60 = 22.01258
      61 = 20.75472
      62 = 20.75472
      63 = 20.44025
      64 = 21.06918
      65 = 20.12579
      66 = 20.75472
      67 = 20.44025
      68 = 21.38365
      69 = 22.01258
      70 = 19.81132
      71 = 20.44025
      72 = 21.06918
      73 = 19.81132
      74 = 19.49686
      75 = 20.44025


#best is k = 30, acc = 31.44654
uKNNval <- union(knnVal.18, res$validate$RPE)
tKNNval <- table(factor(knnVal.18, uKNNval), factor(res$validate$RPE, uKNNval))
confusionMatrix(tKNNval)

Confusion Matrix and Statistics


      4  6  5  7  3  2  8  1 10 7.5  9 5.5 3.5
  4  17 13 14  6 11  1  7  2  0   1  1   0   1
  6  17 11 15 19  3  5  6  5  2   0  1   1   0
  5   9 13 12  6 10  4  5  1  0   1  0   0   0
  7   4  4  7  3  2  0  0  0  0   1  0   0   0
  3   5 12 11  4  4  2  2  3  0   0  1   0   0
  2   4  3  4  1  2  0  2  2  1   0  0   0   0
  8   0  1  0  1  5  0  1  1  0   0  0   0   0
  1   0  0  1  0  3  1  0  0  0   0  0   0   0
  10  0  0  0  0  0  0  0  0  0   0  0   0   0
  7.5 0  0  0  0  0  0  0  0  0   0  0   0   0
  9   0  0  0  0  0  0  0  0  0   0  0   0   0
  5.5 0  0  0  0  0  0  0  0  0   0  0   0   0
  3.5 0  0  0  0  0  0  0  0  0   0  0   0   0

Overall Statistics

               Accuracy : 0.1509
                 95% CI : (0.1134, 0.1951)
    No Information Rate : 0.2013
    P-Value [Acc > NIR] : 0.9913

                  Kappa : -0.0088

 Mcnemar's Test P-Value : NA

Statistics by Class:
```

```
                     Class: 4 Class: 6 Class: 5 Class: 7 Class: 3 Class: 2
Sensitivity           0.30357  0.19298  0.18750 0.075000  0.10000  0.00000
Specificity           0.78244  0.71648  0.80709 0.935252  0.85612  0.93770
Pos Pred Value        0.22973  0.12941  0.19672 0.142857  0.09091  0.00000
Neg Pred Value        0.84016  0.80258  0.79767 0.875421  0.86861  0.95652
Prevalence            0.17610  0.17925  0.20126 0.125786  0.12579  0.04088
Detection Rate        0.05346  0.03459  0.03774 0.009434  0.01258  0.00000
Detection Prevalence  0.23270  0.26730  0.19182 0.066038  0.13836  0.05975
Balanced Accuracy     0.54301  0.45473  0.49729 0.505126  0.47806  0.46885
                     Class: 8 Class: 1 Class: 10 Class: 7.5 Class: 9 Class: 5.5
Sensitivity          0.043478  0.00000  0.000000   0.000000 0.000000   0.000000
Specificity          0.972881  0.98355  1.000000   1.000000 1.000000   1.000000
Pos Pred Value       0.111111  0.00000       NaN        NaN      NaN        NaN
Neg Pred Value       0.928803  0.95527  0.990566   0.990566 0.990566   0.996855
Prevalence           0.072327  0.04403  0.009434   0.009434 0.009434   0.003145
Detection Rate       0.003145  0.00000  0.000000   0.000000 0.000000   0.000000
Detection Prevalence 0.028302  0.01572  0.000000   0.000000 0.000000   0.000000
Balanced Accuracy    0.508180  0.49178  0.500000   0.500000 0.500000   0.500000
                     Class: 3.5
Sensitivity            0.000000
Specificity            1.000000
Pos Pred Value              NaN
Neg Pred Value         0.996855
Prevalence             0.003145
Detection Rate         0.000000
Detection Prevalence   0.000000
Balanced Accuracy      0.500000
```

```
########################################

dataEx <- data.frame(col1=runif(20), col2=runif(20),
                     col3=runif(20), col4=runif(20), col5=runif(20))
bootControl <- trainControl(number = 1)
knnGrid <- expand.grid(.k=c(2:5))
set.seed(42)
knnFit1 <- train(dataEx[,-c(1)], dataEx[,1], method = "knn", trControl = bootControl,
                 verbose = FALSE, tuneGrid = knnGrid )
knnFit1

k-Nearest Neighbors

20 samples
 4 predictor

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 20
Resampling results across tuning parameters:

  k  RMSE       Rsquared     MAE
  2  0.3403888  0.115277660  0.2771437
  3  0.3280267  0.099286255  0.2693268
  4  0.2534202  0.003126126  0.2090370
  5  0.2715250  0.012984973  0.2186794

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was k = 4.

knnFit1$results
```

```
  k      RMSE    Rsquared       MAE RMSESD RsquaredSD MAESD
1 2 0.3403888 0.115277660 0.2771437     NA         NA    NA
2 3 0.3280267 0.099286255 0.2693268     NA         NA    NA
3 4 0.2534202 0.003126126 0.2090370     NA         NA    NA
4 5 0.2715250 0.012984973 0.2186794     NA         NA    NA
```

```
knnFit1$bestTune
```

```
  k
3 4
```

```
knnFit1$finalModel
```

```
4-nearest neighbor regression model
```

```
knnFit1$results$Rsquared
```

```
[1] 0.115277660 0.099286255 0.003126126 0.012984973
```

```
knnFit1.sorted <- knnFit1$results[order(knnFit1$results$Rsquared),]
knnFit1.sorted
```

```
  k      RMSE    Rsquared       MAE RMSESD RsquaredSD MAESD
3 4 0.2534202 0.003126126 0.2090370     NA         NA    NA
4 5 0.2715250 0.012984973 0.2186794     NA         NA    NA
2 3 0.3280267 0.099286255 0.2693268     NA         NA    NA
1 2 0.3403888 0.115277660 0.2771437     NA         NA    NA
```

```
knnFit1.sorted[1,'Rsquared']
```

```
[1] 0.003126126
```

```
#New column id

#all possoble regression
#press, r^2


#lmTest <- MSpred(RPE ~ Distance + `Max Speed` + `HML Distance` + `HML Efforts` + `Sprint Distance`, data = te
#lmTest2 #<- MSpred(other models)
#tset3

#lmValidate <- MSpred(test2, data = valiate)

kTrain <- ols_step_forward_aic(lmTrain)
kTrain
```

```
                                Selection Summary
-------------------------------------------------------------------------------------
Variable                         AIC        Sum Sq       RSS        R-Sq      Adj. R-Sq
-------------------------------------------------------------------------------------
`Average Metabolic Power`    3573.527     989.934    2357.149    0.29576       0.29502
Decelerations                3494.414    1182.258    2164.825    0.35322       0.35186
`HML Density`                3471.546    1238.018    2109.065    0.36988       0.36789
`Dynamic Stress Load`        3449.914    1289.674    2057.409    0.38531       0.38272
Impacts                      3444.454    1305.715    2041.367    0.39011       0.38689
`Max Speed`                  3441.763    1315.739    2031.344    0.39310       0.38925
```

```
Accelerations                3440.713    1322.232    2024.851    0.39504      0.39056
`HML Efforts`                3439.720    1328.579    2018.504    0.39694      0.39183
Distance                     3435.281    1342.173    2004.910    0.40100      0.39528
Duration                     3429.807    1357.834    1989.249    0.40568      0.39937
`Speed Intensity`            3422.228    1377.729    1969.354    0.41162      0.40474
`HML Distance`               3407.718    1411.554    1935.529    0.42173      0.41434
-------------------------------------------------------------------------------------
```

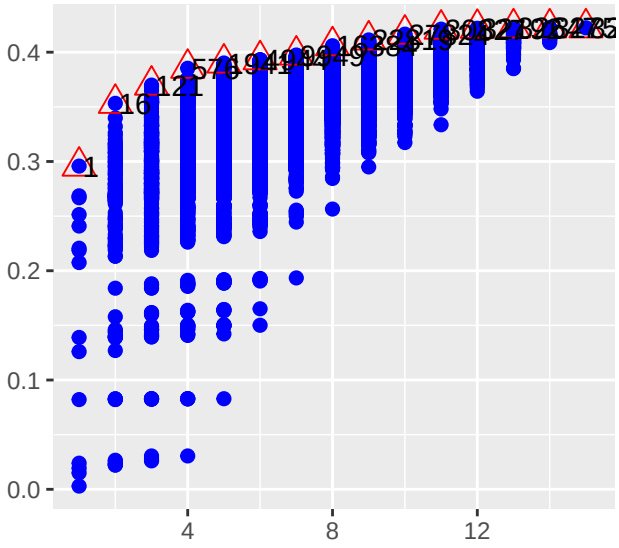
```
kTest <- ols_step_forward_aic(lmTest) #avg metabolic, avg HR, and decelerations
kTest
```

```
                                 Selection Summary
-------------------------------------------------------------------------------------------
Variable                                AIC       Sum Sq       RSS       R-Sq      Adj. R-Sq
-------------------------------------------------------------------------------------------
`HML Distance`                     1150.666    392.082    681.130    0.36533       0.36333
`HML Density`                      1136.516    425.810    647.402    0.39676       0.39293
Decelerations                      1114.116    473.627    599.584    0.44132       0.43598
`Dynamic Stress Load`              1099.247    504.604    568.607    0.47018       0.46341
Sprints                            1089.218    525.711    547.500    0.48985       0.48167
`Average Metabolic Power`          1084.063    537.892    535.320    0.50120       0.49157
Accelerations                      1081.547    545.440    527.771    0.50823       0.49713
`Speed Intensity`                  1079.720    551.754    521.457    0.51412       0.50154
Duration                           1072.684    566.364    506.848    0.52773       0.51393
`High Speed Running (Relative)`    1072.126    570.423    502.788    0.53151       0.51625
-------------------------------------------------------------------------------------------
```

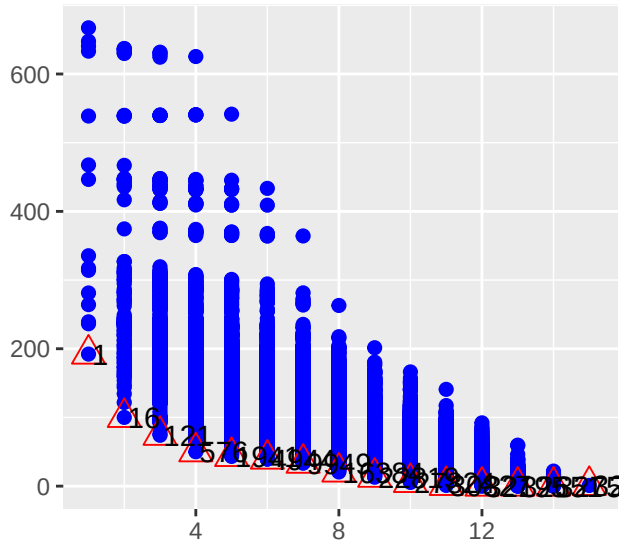
```
#kNew <- ols_step_forward_aic(lm1)
#kNew
```

```
#First Level: Training Data
##All possible regressions
allReg <- ols_step_all_possible(lmTrain)
plot(allReg)
```

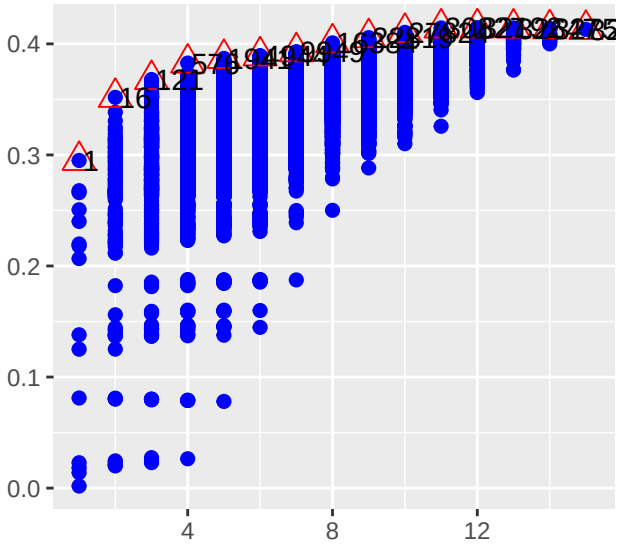
R−Square



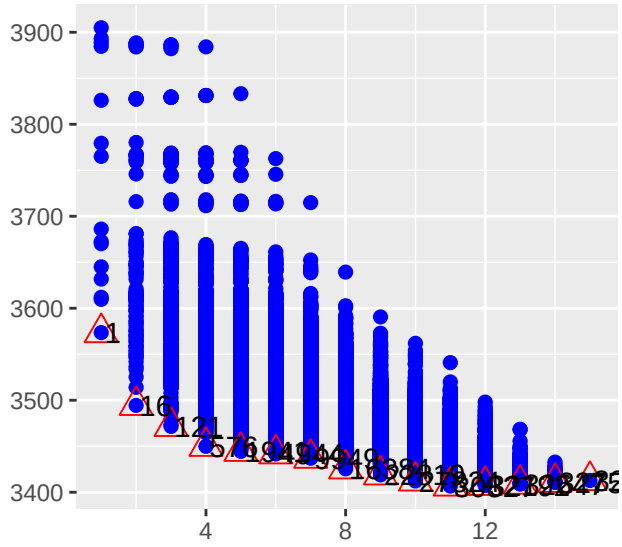
Cp

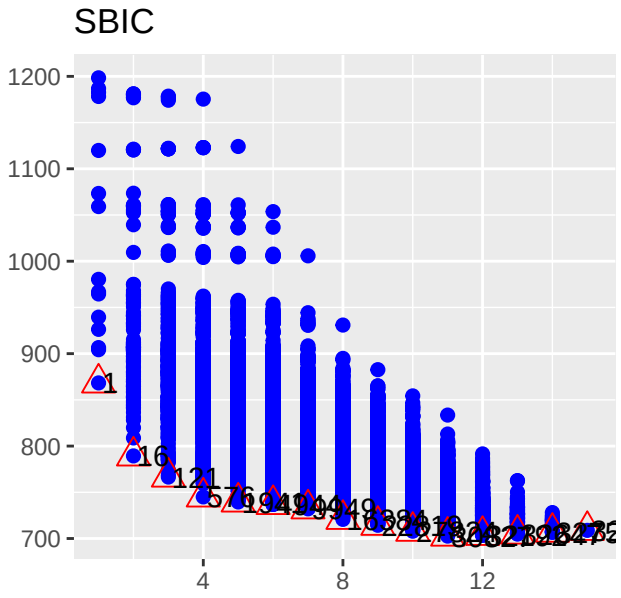


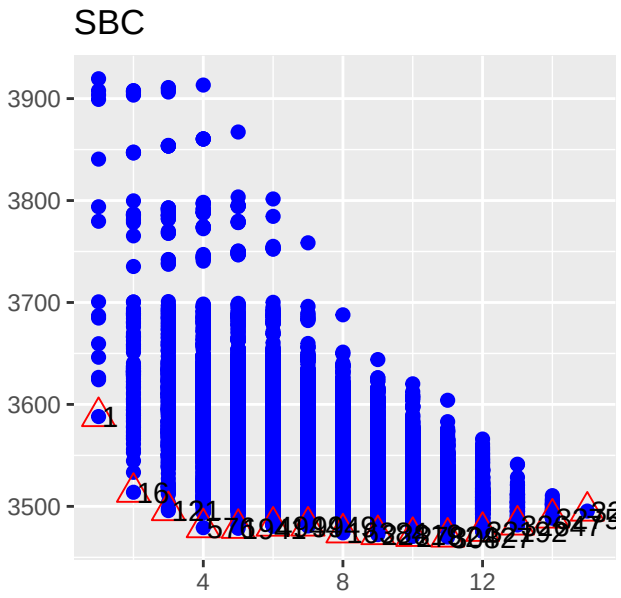
Adj. R−Square



AIC








```
allReg$model
```

```
NULL
```

```
##Compare in terms of PRESS
#Press1 func was here

#PRESS(allReg) #3571, AIC() = 6041, BIC() = 6150 #######For all possiblle regressions

# #Sort based on adj r^2, cp, etc. for vars
#
# #Based on adj R^2
# #The top6 regressors are: avg metabolic power, speed intensity, distance, hml distance, decelerations, hml e
#
# #Second Level:
# ################################################
# #Splitting into 3 datasets
# splits = c(train = .6, test = .2, validate = .2)
```

```
# ################################################
# #or should all possible be using train
# lmTrain <- lm(RPE ~ Distance + `Max Speed` + `HML Distance` + `HML Efforts` + `Sprint Distance` + Sprints +
#                 Accelerations + Decelerations + `Average Heart Rate` + `Max Heart Rate` +
#                 `Average Metabolic Power` + `Dynamic Stress Load` + `Heart Rate Exertion` +
#                 `High Speed Running (Relative)` + `HML Density` + `Speed Intensity` +
#                 Impacts + Duration, data = res$train)
# #also test it to compare
# ################################################
#
# #forward1 <- ols_step_forward_aic(RPE ~ all 19 models, data = res$train) #with all other models with all k's
# #forward1 #avg metabolic, deceleration, hml density, avg hr, distance, max HR  off of AIC
# #plot(forward1)
# #forward1$model
# #based on adj R^2, AIC, malloc cp, bic
#
# ##not yetmod_error(forward1) #comparing the MSpred for the three diffferent models from forward AIC
# #not yet Then take model with best MSpred, and use on validate
#
#
#
# #Repeat same steps using all 19 regressors for train data for: 2 model cp / adj r2
# ##Backwards AIC
# ##Stepwise AIC
# ##KNN
# ##Boosted Forests
# ##Normal GLM
# ##Poisson Regression
# ##Penalized Regression
# ##Ridge
# ##Lasso
# ##Elastic Net
# ##SVM
#
#
# #Then do top k for each approach, use on test
# #MSpred and problem children - combined
#
# #avg metabolic, deceleration, hml density
#
# #Winner - KNN with its k Regressors and use that on validation dataset
# #MSpred and problem children - how is it changing, confirming if similar
#
#
#
#
#
#
#
# ###Where each has a different model of regressors
# ###And the best model is chosen by best MSpred, also dealing with problem children
# ###Then take that best model and use it against the validation data set
#
# #Compute MSpred of each validation for each of the best models
# #Compare which approach has the MSpred,
# #Conclusion: that model and approach is best for predictions


endOverall <- Sys.time()
overallTime <- endOverall - startOverall
overallTime
```

```
Time difference of 1.669532 hours
```