527 project Demo

2024-12-05

Contents

Abstract 1
Model Choice: Full Model and Null Model with Stepwise Selection 4
transformation and diagnostic plots 14
outlier and influential points 20
colinearity 27
LASSO, Ridge, and Elastic Net Regression 29
Conclusion 36

Abstract

This study explores the relationship between calorie expenditure and various factors such as age, gender, workout type, session duration, and heart rate among gym enthusiasts. Using regression analysis and random forest modeling, we developed predictive models to estimate calorie burn based on these variables. The analysis highlights key factors influencing calorie expenditure and provides insights into optimizing workout routines for individual fitness goals. Our findings offer practical guidance for fitness enthusiasts and professionals to design personalized and efficient exercise programs. ## analysis the dataset

```r
gym_data <- read.csv("gym_data.csv")

# Display the first few rows of the data
head(gym_data)
```

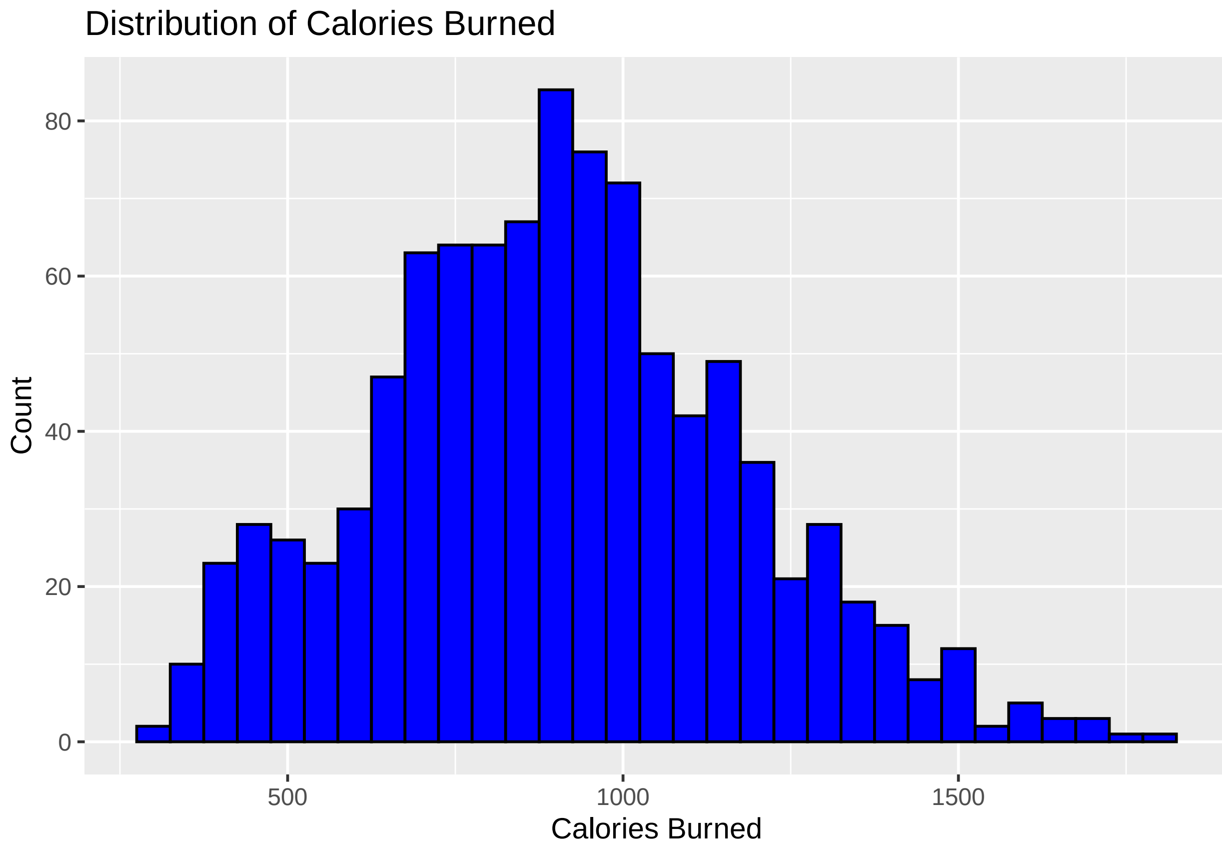
```r
# Check the structure of the dataset
str(gym_data)
```

```
## 'data.frame':    973 obs. of  15 variables:
##  $ Age                          : int  56 46 32 25 38 56 36 40 28 28 ...
##  $ Gender                       : chr  "Male" "Female" "Female" "Male" ...
##  $ Weight..kg.                  : num  88.3 74.9 68.1 53.2 46.1 ...
##  $ Height..m.                   : num  1.71 1.53 1.66 1.7 1.79 1.68 1.72 1.51 1.94 1.84 ...
##  $ Max_BPM                      : int  180 179 167 190 188 168 174 189 185 169 ...
##  $ Avg_BPM                      : int  157 151 122 164 158 156 169 141 127 136 ...
##  $ Resting_BPM                  : int  60 66 54 56 68 74 73 64 52 64 ...
##  $ Session_Duration..hours.     : num  1.69 1.3 1.11 0.59 0.64 1.59 1.49 1.27 1.03 1.08 ...
##  $ Calories_Burned              : num  1313 883 677 532 556 ...
##  $ Workout_Type                 : chr  "Yoga" "HIIT" "Cardio" "Strength" ...
##  $ Fat_Percentage               : num  12.6 33.9 33.4 28.8 29.2 15.5 21.3 30.6 28.9 29.7 ...
##  $ Water_Intake..liters.        : num  3.5 2.1 2.3 2.1 2.8 2.7 2.3 1.9 2.6 2.7 ...
##  $ Workout_Frequency..days.week.: int  4 4 4 3 3 5 3 3 4 3 ...
##  $ Experience_Level             : int  3 2 2 1 1 3 2 2 2 1 ...
##  $ BMI                          : num  30.2 32 24.7 18.4 14.4 ...
```

```r
# Provide summary statistics of the dataset
summary(gym_data)
```

```
##       Age           Gender           Weight..kg.      Height..m.
##  Min.   :18.00   Length:973         Min.   : 40.00   Min.   :1.500
##  1st Qu.:28.00   Class :character   1st Qu.: 58.10   1st Qu.:1.620
##  Median :40.00   Mode  :character   Median : 70.00   Median :1.710
##  Mean   :38.68                      Mean   : 73.85   Mean   :1.723
##  3rd Qu.:49.00                      3rd Qu.: 86.00   3rd Qu.:1.800
##  Max.   :59.00                      Max.   :129.90   Max.   :2.000
##     Max_BPM         Avg_BPM       Resting_BPM    Session_Duration..hours.
##  Min.   :160.0   Min.   :120.0   Min.   :50.00   Min.   :0.500
##  1st Qu.:170.0   1st Qu.:131.0   1st Qu.:56.00   1st Qu.:1.040
##  Median :180.0   Median :143.0   Median :62.00   Median :1.260
##  Mean   :179.9   Mean   :143.8   Mean   :62.22   Mean   :1.256
##  3rd Qu.:190.0   3rd Qu.:156.0   3rd Qu.:68.00   3rd Qu.:1.460
##  Max.   :199.0   Max.   :169.0   Max.   :74.00   Max.   :2.000
##  Calories_Burned  Workout_Type       Fat_Percentage  Water_Intake..liters.
##  Min.   : 303.0   Length:973         Min.   :10.00   Min.   :1.500
##  1st Qu.: 720.0   Class :character   1st Qu.:21.30   1st Qu.:2.200
##  Median : 893.0   Mode  :character   Median :26.20   Median :2.600
##  Mean   : 905.4                      Mean   :24.98   Mean   :2.627
##  3rd Qu.:1076.0                      3rd Qu.:29.30   3rd Qu.:3.100
##  Max.   :1783.0                      Max.   :35.00   Max.   :3.700
##  Workout_Frequency..days.week. Experience_Level      BMI
##  Min.   :2.000                 Min.   :1.00     Min.   :12.32
##  1st Qu.:3.000                 1st Qu.:1.00     1st Qu.:20.11
##  Median :3.000                 Median :2.00     Median :24.16
##  Mean   :3.322                 Mean   :1.81     Mean   :24.91
##  3rd Qu.:4.000                 3rd Qu.:2.00     3rd Qu.:28.56
##  Max.   :5.000                 Max.   :3.00     Max.   :49.84
```

```r
library(ggplot2)

ggplot(gym_data, aes(x = Calories_Burned)) +
  geom_histogram(binwidth = 50, fill = "blue", color = "black") +
  labs(title = "Distribution of Calories Burned", x = "Calories Burned", y = "Count")
```




```r
numeric_vars <- gym_data[, sapply(gym_data, is.numeric)]
correlations <- cor(numeric_vars, use = "complete.obs")
correlations["Calories_Burned", ]
```

```
##                           Age                  Weight..kg.
##                  -0.154678760                  0.095443473
##                    Height..m.                      Max_BPM
##                   0.086348051                  0.002090016
##                       Avg_BPM                  Resting_BPM
##                   0.339658667                  0.016517951
##       Session_Duration..hours.              Calories_Burned
##                   0.908140376                  1.000000000
##                Fat_Percentage       Water_Intake..liters.
##                  -0.597615248                  0.356930683
## Workout_Frequency..days.week.             Experience_Level
##                   0.576150125                  0.694129448
##                           BMI
##                   0.059760826
```

After loading the dataset and performing some initial exploration, we observed the following:

The histogram indicates that Calories_Burned has a unimodal distribution with a slight skew to the right. Most values fall within the range of 800 to 1400. This suggests that most observations are clustered in this range, but there are a few higher values that might require further attention for outliers or special cases.

By calculating the correlation between Calories_Burned and other numeric variables, we identified the strength of their linear relationships. Strongly correlated variables (positive or negative) might have more predictive power for our target variable, while weak correlations might indicate limited predictive value.

Next Step: fit a full model to understand the relationships between Calories_Burned and other variables in the dataset, we will start with a full model that includes all predictors

Model Choice: Full Model and Null Model with Stepwise Selection

To develop a predictive model for `Calories_Burned`, both a **full model** and a **null model** were used as starting points for **stepwise selection** to find the optimal set of predictors.

Steps:

1. **Data Split**: The dataset was divided into 70% training data and 30% testing data for model training and evaluation.
2. **Full Model**:
   - The full model included all predictors to assess their collective contribution to explaining `Calories_Burned`.
   - This model served as the upper limit for the stepwise selection process.
3. **Null Model**:
   - The null model only included the intercept, assuming no predictors were significant.
   - This model served as the lower limit for stepwise selection.
4. **Stepwise Selection**:
   - **BIC**: Stepwise selection based on the Bayesian Information Criterion identified a parsimonious model with fewer predictors by strongly penalizing model complexity.
   - **AIC**: Stepwise selection based on the Akaike Information Criterion provided a more flexible approach, potentially retaining more predictors for better predictive performance.

Conclusion: Using both the full and null models in stepwise selection ensures a systematic approach to identifying the optimal subset of predictors. BIC favors simpler models, while AIC allows for slightly more complexity, providing a balance between interpretability and predictive accuracy.

```r
# devide the dataset to training and testing
set.seed(123)
train_index <- sample(1:nrow(gym_data), 0.7 * nrow(gym_data))
train_data <- gym_data[train_index, ]
test_data <- gym_data[-train_index, ]
# Fit a full model
full_model <- lm(Calories_Burned ~ ., data = train_data)
summary(full_model)
```

```
##
## Call:
## lm(formula = Calories_Burned ~ ., data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -113.713  -25.164   -2.278   24.170  169.962
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -1.154e+03  1.038e+02 -11.112  < 2e-16 ***
## Age                          -3.483e+00  1.249e-01 -27.888  < 2e-16 ***
## GenderMale                    8.050e+01  5.410e+00  14.878  < 2e-16 ***
## Weight..kg.                  -1.750e+00  5.985e-01  -2.924 0.003574 **
```

```
## Height..m.                    1.898e+02  5.542e+01   3.425 0.000654 ***
## Max_BPM                       2.315e-02  1.327e-01   0.174 0.861566
## Avg_BPM                       6.349e+00  1.047e-01  60.629  < 2e-16 ***
## Resting_BPM                   3.416e-01  2.126e-01   1.607 0.108589
## Session_Duration..hours.      7.128e+02  7.079e+00 100.699  < 2e-16 ***
## Workout_TypeHIIT              3.786e-01  4.297e+00   0.088 0.929805
## Workout_TypeStrength         -2.641e-01  4.181e+00  -0.063 0.949662
## Workout_TypeYoga             -5.501e+00  4.294e+00  -1.281 0.200687
## Fat_Percentage               -4.977e-01  3.908e-01  -1.273 0.203297
## Water_Intake..liters.        -3.168e+00  3.785e+00  -0.837 0.402852
## Workout_Frequency..days.week.  2.457e+00  3.030e+00   0.811 0.417712
## Experience_Level             -2.820e+00  4.748e+00  -0.594 0.552715
## BMI                           5.580e+00  1.821e+00   3.064 0.002270 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.16 on 664 degrees of freedom
## Multiple R-squared:  0.9802, Adjusted R-squared:  0.9797
## F-statistic:  2050 on 16 and 664 DF,  p-value: < 2.2e-16
```

```r
#if when we have a large number of predictors, according to the summary, we can see that some predictor

# Null model with only the intercept
null_model <- lm(Calories_Burned ~ 1, data = train_data)

# Stepwise selection both directiona
# Number of observations
n <- nrow(train_data)

# Stepwise selection based on BIC
stepwise_model_bic <- step(null_model,scope = list(lower = null_model, upper = full_model), direction =
```

```
## Start:  AIC=7653.93
## Calories_Burned ~ 1
##
##                                 Df Sum of Sq      RSS    AIC
## + Session_Duration..hours.       1  41918119  9383194 6503.6
## + Experience_Level               1  23978491 27322823 7231.4
## + Fat_Percentage                 1  18131333 33169980 7363.5
## + Workout_Frequency..days.week.  1  16513112 34788202 7395.9
## + Water_Intake..liters.          1   6431164 44870149 7569.2
## + Avg_BPM                        1   6406738 44894575 7569.6
## + Age                            1   1951897 49349416 7634.0
## + Gender                         1    996655 50304658 7647.1
## <none>                                      51301313 7653.9
## + Weight..kg.                    1    317307 50984006 7656.2
## + Height..m.                     1    304705 50996608 7656.4
## + BMI                            1    112172 51189142 7659.0
## + Resting_BPM                    1     18586 51282727 7660.2
## + Max_BPM                        1      7995 51293318 7660.3
## + Workout_Type                   3    192349 51108965 7670.9
##
## Step:  AIC=6503.57
## Calories_Burned ~ Session_Duration..hours.
##
```

```
##                                 Df Sum of Sq      RSS    AIC
## + Avg_BPM                        1   5702415  3680779 5872.8
## + Gender                         1   1399920  7983275 6400.1
## + Age                            1   1232801  8150393 6414.2
## + Weight..kg.                    1    738661  8644533 6454.3
## + Height..m.                     1    595620  8787574 6465.4
## + Water_Intake..liters.          1    543686  8839509 6469.4
## + Fat_Percentage                 1    327931  9055264 6485.9
## + BMI                            1    266581  9116613 6490.5
## <none>                                        9383194 6503.6
## + Resting_BPM                    1     77708  9305487 6504.4
## + Workout_Frequency..days.week.  1     12399  9370796 6509.2
## + Experience_Level               1     11595  9371599 6509.3
## + Max_BPM                        1      7152  9376042 6509.6
## + Workout_Type                   3     17204  9365991 6521.9
## - Session_Duration..hours.       1  41918119 51301313 7653.9
##
## Step:  AIC=5872.82
## Calories_Burned ~ Session_Duration..hours. + Avg_BPM
##
##                                 Df Sum of Sq      RSS    AIC
## + Gender                         1   1401081  2279698 5553.1
## + Age                            1   1292200  2388579 5584.9
## + Weight..kg.                    1    649206  3031573 5747.2
## + Height..m.                     1    631952  3048827 5751.1
## + Water_Intake..liters.          1    539786  3140993 5771.3
## + Fat_Percentage                 1    368052  3312727 5807.6
## + BMI                            1    197825  3482954 5841.7
## <none>                                        3680779 5872.8
## + Resting_BPM                    1      3556  3677223 5878.7
## + Workout_Frequency..days.week.  1      1401  3679378 5879.1
## + Max_BPM                        1      1139  3679640 5879.1
## + Experience_Level               1       577  3680202 5879.2
## + Workout_Type                   3      5951  3674828 5891.3
## - Avg_BPM                        1   5702415  9383194 6503.6
## - Session_Duration..hours.       1  41213796 44894575 7569.6
##
## Step:  AIC=5553.08
## Calories_Burned ~ Session_Duration..hours. + Avg_BPM + Gender
##
##                                 Df Sum of Sq      RSS    AIC
## + Age                            1   1229305  1050392 5031.9
## <none>                                        2279698 5553.1
## + Weight..kg.                    1     18036  2261661 5554.2
## + Height..m.                     1     15717  2263981 5554.9
## + Water_Intake..liters.          1      9833  2269864 5556.7
## + BMI                            1      4609  2275088 5558.2
## + Resting_BPM                    1      2924  2276773 5558.7
## + Fat_Percentage                 1      1684  2278013 5559.1
## + Max_BPM                        1      1028  2278669 5559.3
## + Workout_Frequency..days.week.  1       123  2279574 5559.6
## + Experience_Level               1       123  2279575 5559.6
## + Workout_Type                   3      9768  2269929 5569.7
## - Gender                         1   1401081  3680779 5872.8
```

```
## - Avg_BPM                        1   5703577  7983275 6400.1
## - Session_Duration..hours.       1  41613983 43893681 7560.8
##
## Step:  AIC=5031.91
## Calories_Burned ~ Session_Duration..hours. + Avg_BPM + Gender +
##     Age
##
##                                Df Sum of Sq      RSS    AIC
## <none>                                       1050392 5031.9
## + Height..m.                    1      4323  1046069 5035.6
## + Resting_BPM                   1      3433  1046959 5036.2
## + Fat_Percentage                1      2429  1047963 5036.9
## + Workout_Frequency..days.week. 1      1917  1048475 5037.2
## + Weight..kg.                   1       951  1049441 5037.8
## + Experience_Level              1       496  1049896 5038.1
## + Water_Intake..liters.         1       340  1050052 5038.2
## + BMI                           1       123  1050269 5038.4
## + Max_BPM                       1        13  1050379 5038.4
## + Workout_Type                  3      3991  1046401 5048.9
## - Age                           1   1229305  2279698 5553.1
## - Gender                        1   1338187  2388579 5584.9
## - Avg_BPM                       1   5761488  6811880 6298.5
## - Session_Duration..hours.      1  40882421 41932814 7536.2
```

```r
summary(stepwise_model_bic)
```

```
##
## Call:
## lm(formula = Calories_Burned ~ Session_Duration..hours. + Avg_BPM +
##     Gender + Age, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -113.943  -26.736   -2.965   24.898  175.614
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -820.8356    16.8076  -48.84   <2e-16 ***
## Session_Duration..hours.  715.5878     4.4116  162.21   <2e-16 ***
## Avg_BPM                     6.3692     0.1046   60.89   <2e-16 ***
## GenderMale                 88.9493     3.0310   29.35   <2e-16 ***
## Age                        -3.4951     0.1243  -28.13   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.42 on 676 degrees of freedom
## Multiple R-squared:  0.9795, Adjusted R-squared:  0.9794
## F-statistic:  8085 on 4 and 676 DF,  p-value: < 2.2e-16
```

```r
stepwise_model_aic <- step(null_model,scope = list(lower = null_model, upper = full_model), direction =
```

```
## Start:  AIC=7649.4
## Calories_Burned ~ 1
##
##                                Df Sum of Sq      RSS    AIC
```

```
## + Session_Duration..hours.       1  41918119  9383194 6494.5
## + Experience_Level               1  23978491 27322823 7222.4
## + Fat_Percentage                 1  18131333 33169980 7354.4
## + Workout_Frequency..days.week.  1  16513112 34788202 7386.9
## + Water_Intake..liters.          1   6431164 44870149 7560.2
## + Avg_BPM                        1   6406738 44894575 7560.6
## + Age                            1   1951897 49349416 7625.0
## + Gender                         1    996655 50304658 7638.0
## + Weight..kg.                    1    317307 50984006 7647.2
## + Height..m.                     1    304705 50996608 7647.3
## <none>                                      51301313 7649.4
## + BMI                            1    112172 51189142 7649.9
## + Resting_BPM                    1     18586 51282727 7651.2
## + Max_BPM                        1      7995 51293318 7651.3
## + Workout_Type                   3    192349 51108965 7652.8
##
## Step:  AIC=6494.52
## Calories_Burned ~ Session_Duration..hours.
##
##                                 Df Sum of Sq      RSS    AIC
## + Avg_BPM                        1   5702415  3680779 5859.2
## + Gender                         1   1399920  7983275 6386.5
## + Age                            1   1232801  8150393 6400.6
## + Weight..kg.                    1    738661  8644533 6440.7
## + Height..m.                     1    595620  8787574 6451.9
## + Water_Intake..liters.          1    543686  8839509 6455.9
## + Fat_Percentage                 1    327931  9055264 6472.3
## + BMI                            1    266581  9116613 6476.9
## + Resting_BPM                    1     77708  9305487 6490.9
## <none>                                       9383194 6494.5
## + Workout_Frequency..days.week.  1     12399  9370796 6495.6
## + Experience_Level               1     11595  9371599 6495.7
## + Max_BPM                        1      7152  9376042 6496.0
## + Workout_Type                   3     17204  9365991 6499.3
## - Session_Duration..hours.       1  41918119 51301313 7649.4
##
## Step:  AIC=5859.24
## Calories_Burned ~ Session_Duration..hours. + Avg_BPM
##
##                                 Df Sum of Sq      RSS    AIC
## + Gender                         1   1401081  2279698 5535.0
## + Age                            1   1292200  2388579 5566.8
## + Weight..kg.                    1    649206  3031573 5729.1
## + Height..m.                     1    631952  3048827 5733.0
## + Water_Intake..liters.          1    539786  3140993 5753.2
## + Fat_Percentage                 1    368052  3312727 5789.5
## + BMI                            1    197825  3482954 5823.6
## <none>                                       3680779 5859.2
## + Resting_BPM                    1      3556  3677223 5860.6
## + Workout_Frequency..days.week.  1      1401  3679378 5861.0
## + Max_BPM                        1      1139  3679640 5861.0
## + Experience_Level               1       577  3680202 5861.1
## + Workout_Type                   3      5951  3674828 5864.1
## - Avg_BPM                        1   5702415  9383194 6494.5
```

```
## - Session_Duration..hours.       1  41213796 44894575 7560.6
##
## Step:  AIC=5534.99
## Calories_Burned ~ Session_Duration..hours. + Avg_BPM + Gender
##
##                                 Df Sum of Sq      RSS    AIC
## + Age                            1   1229305  1050392 5009.3
## + Weight..kg.                    1     18036  2261661 5531.6
## + Height..m.                     1     15717  2263981 5532.3
## + Water_Intake..liters.          1      9833  2269864 5534.0
## <none>                                        2279698 5535.0
## + BMI                            1      4609  2275088 5535.6
## + Resting_BPM                    1      2924  2276773 5536.1
## + Fat_Percentage                 1      1684  2278013 5536.5
## + Max_BPM                        1      1028  2278669 5536.7
## + Workout_Frequency..days.week.  1       123  2279574 5537.0
## + Experience_Level               1       123  2279575 5537.0
## + Workout_Type                   3      9768  2269929 5538.1
## - Gender                         1   1401081  3680779 5859.2
## - Avg_BPM                        1   5703577  7983275 6386.5
## - Session_Duration..hours.       1  41613983 43893681 7547.2
##
## Step:  AIC=5009.3
## Calories_Burned ~ Session_Duration..hours. + Avg_BPM + Gender +
##     Age
##
##                                 Df Sum of Sq      RSS    AIC
## + Height..m.                     1      4323  1046069 5008.5
## + Resting_BPM                    1      3433  1046959 5009.1
## <none>                                        1050392 5009.3
## + Fat_Percentage                 1      2429  1047963 5009.7
## + Workout_Frequency..days.week.  1      1917  1048475 5010.1
## + Weight..kg.                    1       951  1049441 5010.7
## + Experience_Level               1       496  1049896 5011.0
## + Water_Intake..liters.          1       340  1050052 5011.1
## + BMI                            1       123  1050269 5011.2
## + Max_BPM                        1        13  1050379 5011.3
## + Workout_Type                   3      3991  1046401 5012.7
## - Age                            1   1229305  2279698 5535.0
## - Gender                         1   1338187  2388579 5566.8
## - Avg_BPM                        1   5761488  6811880 6280.4
## - Session_Duration..hours.       1  40882421 41932814 7518.1
##
## Step:  AIC=5008.49
## Calories_Burned ~ Session_Duration..hours. + Avg_BPM + Gender +
##     Age + Height..m.
##
##                                 Df Sum of Sq      RSS    AIC
## + Resting_BPM                    1      3794  1042276 5008.0
## <none>                                        1046069 5008.5
## + Fat_Percentage                 1      2371  1043698 5008.9
## + BMI                            1      2276  1043794 5009.0
## + Workout_Frequency..days.week.  1      2062  1044008 5009.1
## - Height..m.                     1      4323  1050392 5009.3
```

```
## + Weight..kg.                    1       941  1045129 5009.9
## + Experience_Level               1       635  1045434 5010.1
## + Water_Intake..liters.          1       362  1045707 5010.3
## + Max_BPM                        1        15  1046055 5010.5
## + Workout_Type                   3      4323  1041746 5011.7
## - Gender                         1    808692  1854762 5396.5
## - Age                            1   1217911  2263981 5532.3
## - Avg_BPM                        1   5764308  6810377 6282.3
## - Session_Duration..hours.       1  40879701 41925770 7520.0
##
## Step:  AIC=5008.01
## Calories_Burned ~ Session_Duration..hours. + Avg_BPM + Gender +
##     Age + Height..m. + Resting_BPM
##
##                                 Df Sum of Sq      RSS    AIC
## <none>                                        1042276 5008.0
## + BMI                            1      2554  1039722 5008.3
## - Resting_BPM                    1      3794  1046069 5008.5
## + Fat_Percentage                 1      2101  1040174 5008.6
## + Workout_Frequency..days.week.  1      1914  1040362 5008.8
## - Height..m.                     1      4683  1046959 5009.1
## + Weight..kg.                    1      1095  1041180 5009.3
## + Experience_Level               1       486  1041789 5009.7
## + Water_Intake..liters.          1       413  1041863 5009.7
## + Max_BPM                        1         4  1042271 5010.0
## + Workout_Type                   3      3914  1038362 5011.5
## - Gender                         1    804636  1846911 5395.6
## - Age                            1   1218134  2260410 5533.2
## - Avg_BPM                        1   5689250  6731526 6276.3
## - Session_Duration..hours.       1  40874044 41916320 7521.8
```

```r
summary(stepwise_model_aic)
```

```
##
## Call:
## lm(formula = Calories_Burned ~ Session_Duration..hours. + Avg_BPM +
##     Gender + Age + Height..m. + Resting_BPM, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -115.53  -26.20   -2.41   25.02  171.09
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -881.9137    32.2916 -27.311   <2e-16 ***
## Session_Duration..hours.  715.9314     4.4036 162.578   <2e-16 ***
## Avg_BPM                     6.3562     0.1048  60.655   <2e-16 ***
## GenderMale                 85.1221     3.7317  22.811   <2e-16 ***
## Age                        -3.4842     0.1241 -28.066   <2e-16 ***
## Height..m.                 25.1815    14.4701   1.740   0.0823 .
## Resting_BPM                 0.3322     0.2121   1.566   0.1178
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.32 on 674 degrees of freedom
```

```
## Multiple R-squared:  0.9797, Adjusted R-squared:  0.9795
## F-statistic:  5417 on 6 and 674 DF,  p-value: < 2.2e-16
```

```r
# Extract the coefficients from the final model
final_model1 <- stepwise_model_bic
final_model_coefficients <- coef(final_model1)
final_model_coefficients
```

```
##            (Intercept) Session_Duration..hours.                  Avg_BPM
##            -820.835603               715.587845                 6.369233
##             GenderMale                      Age
##              88.949297                -3.495094
```

Model Formula-stepwise selection based on BIC

The final linear regression model for predicting `Calories_Burned` is as follows:

$$
\begin{aligned}
\text{Calories\_Burned} = & -820.8356 + 715.5878 \cdot \text{Session\_Duration\_hours}+ \\
& 6.3692 \cdot \text{Avg\_BPM} + 88.9493 \cdot \text{GenderMale}- \\
& 3.4951 \cdot \text{Age}
\end{aligned}
$$

Model Coefficients The table below shows the coefficients estimated by the regression:

Predictor	Coefficient	Std. Error	t-value	p-value	Significance
Intercept	-820.8356	16.8076	-48.84	< 2e-16	***
Session_Duration..hours.	715.5878	4.4116	162.21	< 2e-16	***
Avg_BPM	6.3692	0.1046	60.89	< 2e-16	***
GenderMale	88.9493	3.0310	29.35	< 2e-16	***
Age	-3.4951	0.1243	-28.13	< 2e-16	***

Key Metrics

- **Residual Standard Error (RSE):** 39.42
- **Multiple R-squared:** 0.9795
- **Adjusted R-squared:** 0.9794
- **F-statistic:** 8085 on 4 and 676 degrees of freedom
- **p-value:** < 2.2e-16

Key Metrics

- **R-squared:** 0.9795
- **Adjusted R-squared:** 0.9794
- **Residual Standard Error:** 39.42
- **F-statistic (p-value):** 8085 (< 2.2e-16)

Interpretation

1. **Session Duration:** Each additional hour burns **715.6 more calories**.
2. **Avg_BPM:** A 1-unit increase burns **6.37 additional calories**.
3. **Gender (Male):** Males burn **88.95 more calories** than females.
4. **Age:** Each additional year decreases calorie burn by **3.5 calories**.

Conclusion This model explains 97.95% of the variance in `Calories_Burned` and identifies session duration, average BPM, gender, and age as significant predictors. It provides a reliable tool for estimating calorie expenditure and optimizing workout strategies.

Model Formula-stepwise selection based on AIC

```
# Extract the coefficients from the final model
final_model2 <- stepwise_model_aic
final_model_coefficients <- coef(final_model2)
final_model_coefficients
```

```
##              (Intercept) Session_Duration..hours.                  Avg_BPM
##             -881.9136772              715.9314060                6.3561892
##               GenderMale                      Age               Height..m.
##               85.1221175               -3.4842386               25.1815303
##              Resting_BPM
##                0.3322244
```

Model Formula The AIC-optimized linear regression model for predicting `Calories_Burned` is:

$$
\begin{aligned}
\text{Calories\_Burned} = & -881.9137 + 715.9314 \cdot \text{Session\_Duration\_hours} + \\
& 6.3562 \cdot \text{Avg\_BPM} + 85.1221 \cdot \text{GenderMale} - \\
& 3.4842 \cdot \text{Age} + 25.1815 \cdot \text{Height\_m} + \\
& 0.3322 \cdot \text{Resting\_BPM}
\end{aligned}
$$

Coefficients

Predictor	Coefficient	Std. Error	p-value	Significance
Intercept	-881.9137	32.2916	$< 2e-16$	***
Session_Duration..hours.	715.9314	4.4036	$< 2e-16$	***
Avg_BPM	6.3562	0.1048	$< 2e-16$	***
GenderMale	85.1221	3.7317	$< 2e-16$	***
Age	-3.4842	0.1241	$< 2e-16$	***
Height..m.	25.1815	14.4701	0.0823	.
Resting_BPM	0.3322	0.2121	0.1178	

Key Results

- **R-squared:** 0.9797
- **Residual Standard Error:** 39.32
- **F-statistic:** 5417 ($p < 2.2e-16$)

Significant Predictors

1. **Session Duration**: Each additional hour increases calories burned by **715.93**.
2. **Avg BPM**: Each unit increase adds **6.36 calories**.
3. **Gender (Male)**: Males burn **85.12 more calories** than females.
4. **Age**: Each additional year reduces calories burned by **3.48**.

Conclusion The AIC-optimized model explains **97.97%** of the variance in calories burned. Key predictors like session duration, heart rate, gender, and age provide actionable insights for optimizing fitness plans.

Determine the best model And then we can use the BIC and AIC value to determine the model

```r
# Calculate AIC and BIC for the final model
aic_value1 <- AIC(stepwise_model_bic)
bic_value1 <- BIC(stepwise_model_bic)
aic_value2 <- AIC(stepwise_model_aic)
bic_value2 <- BIC(stepwise_model_aic)

cat("The AIC of stepwise_model_bic:", aic_value1, "\n")
```

```
## The AIC of stepwise_model_bic: 6943.891
```

```r
cat("The BIC of stepwise_model_bic:", bic_value1, "\n")
```

```
## The BIC of stepwise_model_bic: 6971.033
```

```r
cat("The AIC of stepwise_model_aic:", aic_value2, "\n")
```

```
## The AIC of stepwise_model_aic: 6942.609
```

```r
cat("The BIC of stepwise_model_aic:", bic_value2, "\n")
```

```
## The BIC of stepwise_model_aic: 6978.797
```

Results:

- **Stepwise Model (BIC)**:
  - AIC: 6943.891
  - BIC: 6971.033
- **Stepwise Model (AIC)**:
  - AIC: 6942.609
  - BIC: 6978.797

Why Choose AIC? We selected the AIC-based model because it prioritizes **predictive accuracy**. While the BIC-based model slightly simplifies the model by penalizing complexity more heavily, the AIC-based model is more suited for maximizing prediction performance. The lower AIC value (6942.609) supports the choice of the AIC model.

Theoretical Background: AIC and BIC

1. **AIC (Akaike Information Criterion)** The AIC evaluates a model's goodness of fit while penalizing complexity. The formula is:

$$\text{AIC} = -2 \cdot \text{Log-Likelihood} + 2k$$

Where: - Log-Likelihood: Measures how well the model fits the data. - k: Number of estimated parameters in the model.

- **Interpretation**: Lower AIC values indicate a better trade-off between fit and complexity. AIC is more flexible and allows slightly more parameters to improve predictive accuracy.

2. **BIC (Bayesian Information Criterion)** The BIC similarly balances fit and complexity but applies a stronger penalty for additional predictors. The formula is:

$$\text{BIC} = -2 \cdot \text{Log-Likelihood} + k \cdot \log(n)$$

Where: - n: Number of observations. - k: Number of estimated parameters in the model.

- **Interpretation**: Lower BIC values indicate a better model. BIC tends to favor simpler models and is more conservative than AIC.

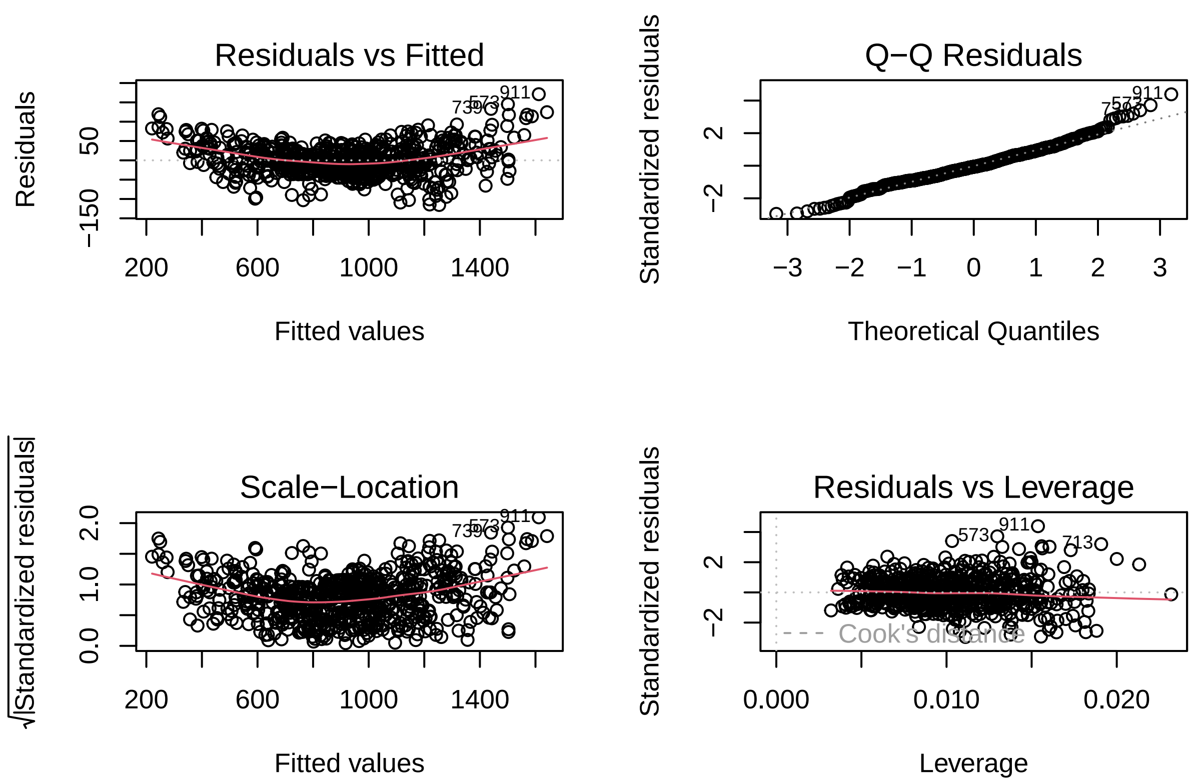
transformation and diagnostic plots

we want a better prediction, so we decide to use the AIC model.

```r
# Extract the coefficients from the final model
final_model <- stepwise_model_aic
final_model_coefficients <- coef(final_model)
final_model_coefficients
```

```
##              (Intercept) Session_Duration..hours.                  Avg_BPM
##             -881.9136772              715.9314060                6.3561892
##               GenderMale                      Age                Height..m.
##               85.1221175               -3.4842386               25.1815303
##              Resting_BPM
##                0.3322244
```

```r
# Diagnostic plots
par(mfrow = c(2, 2))
plot(final_model)
```



The diagnostic plots help us assess the assumptions of the linear regression model. We can check for linearity, homoscedasticity, normality of residuals, and influential points. If any of these assumptions are violated, we may need to address them before interpreting the model results.

Non-Linearity: The "Residuals vs Fitted" plot shows a curved pattern, suggesting a non-linear relationship between the predictors and the response.

Non-Normality: The "Normal Q-Q" plot shows deviations at the tails, indicating the residuals are not perfectly normal.

Heteroscedasticity: The "Scale-Location" plot shows a slight pattern, indicating that the variance of residuals may not be constant.

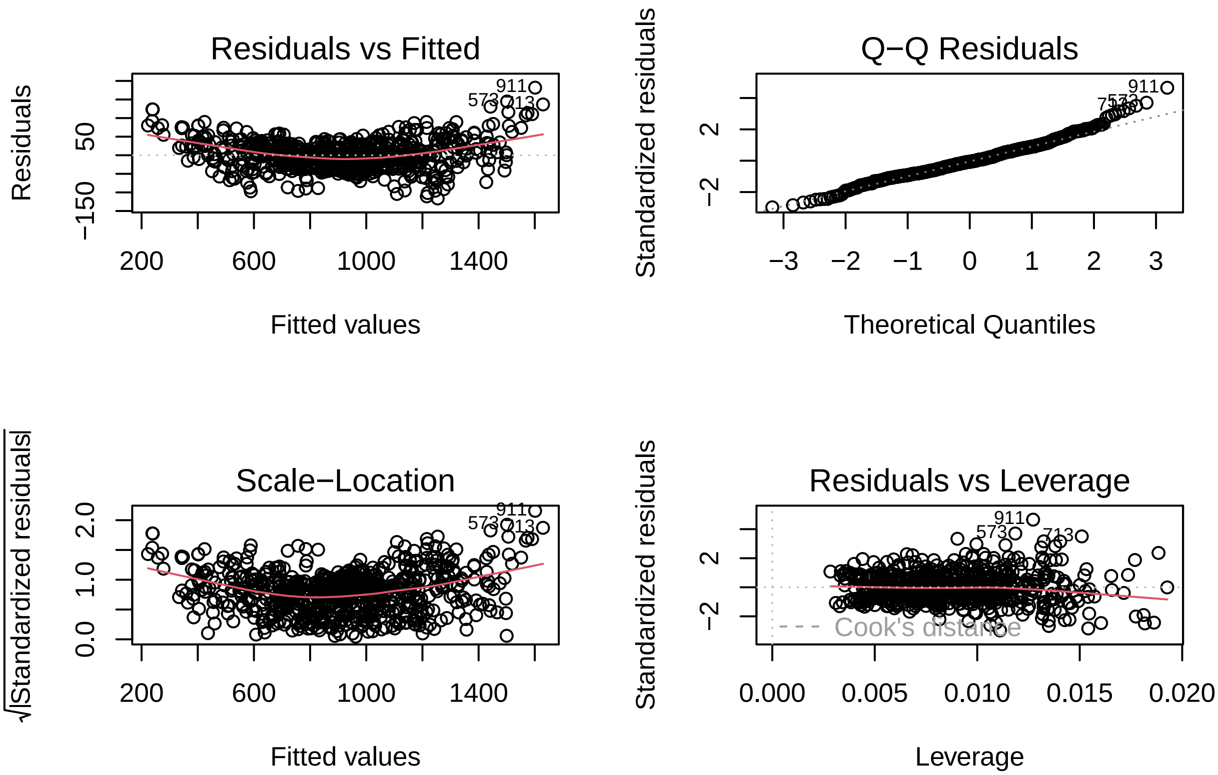
To address these issues, we can try transforming the predictors or the response variable to linearize relationships, stabilize variance, and normalize residuals.

Add log transformation to the model

```r
# Log transformation of the response
log_model <- lm(Calories_Burned ~ Session_Duration..hours. + log(Avg_BPM) +
                  Gender + Age + log(Resting_BPM), data = train_data)
summary(log_model)
```

```
##
## Call:
## lm(formula = Calories_Burned ~ Session_Duration..hours. + log(Avg_BPM) +
##     Gender + Age + log(Resting_BPM), data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -116.248  -26.843   -2.657   23.422  181.829
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -4508.068     87.959 -51.252   <2e-16 ***
## Session_Duration..hours.   715.524      4.402 162.532   <2e-16 ***
## log(Avg_BPM)               911.802     15.034  60.649   <2e-16 ***
## GenderMale                  88.797      3.024  29.366   <2e-16 ***
## Age                         -3.509      0.124 -28.306   <2e-16 ***
## log(Resting_BPM)            18.963     13.011   1.457    0.145
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.32 on 675 degrees of freedom
## Multiple R-squared:  0.9797, Adjusted R-squared:  0.9795
## F-statistic:  6500 on 5 and 675 DF,  p-value: < 2.2e-16
```

```r
par(mfrow = c(2, 2))
plot(log_model)
```



The log transformation improved model validity by addressing key issues with linearity, normality, and variance consistency. However, further investigation of influential points and potential additional transformations is recommended to fully optimize the model.

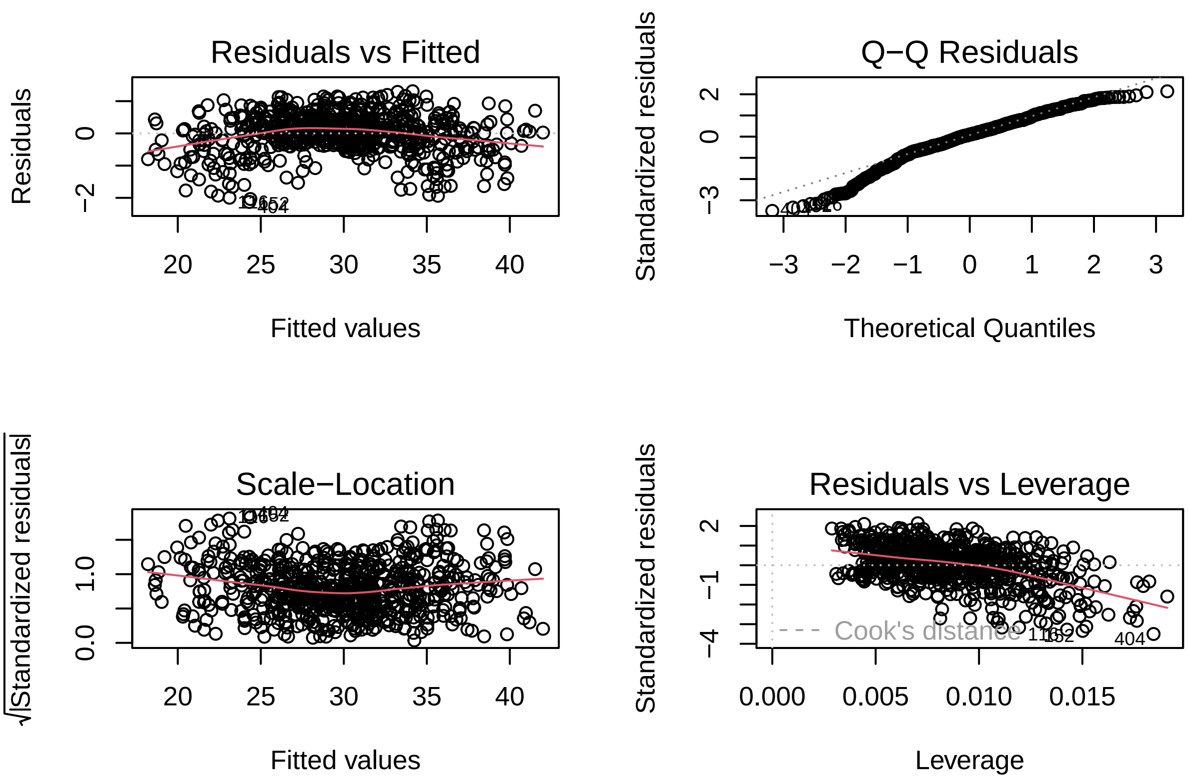
Add square root transformation to the model

```r
# Square root transformation of response
sqrt_model <- lm(sqrt(Calories_Burned) ~ Session_Duration..hours. + Avg_BPM +
                   Gender + Age + Resting_BPM, data = train_data)
summary(sqrt_model)
```

```
##
## Call:
## lm(formula = sqrt(Calories_Burned) ~ Session_Duration..hours. +
##     Avg_BPM + Gender + Age + Resting_BPM, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.12821 -0.32294  0.06018  0.41394  1.30784
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)              7.633e-01  3.211e-01   2.377   0.0177 *
## Session_Duration..hours. 1.213e+01  6.870e-02 176.613   <2e-16 ***
## Avg_BPM                  1.053e-01  1.635e-03  64.417   <2e-16 ***
## GenderMale               1.446e+00  4.719e-02  30.654   <2e-16 ***
## Age                     -5.649e-02  1.935e-03 -29.199   <2e-16 ***
```

```
## Resting_BPM                 9.879e-05  3.307e-03   0.030   0.9762
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6137 on 675 degrees of freedom
## Multiple R-squared:  0.9825, Adjusted R-squared:  0.9824
## F-statistic:  7591 on 5 and 675 DF,  p-value: < 2.2e-16
```

```
par(mfrow = c(2, 2))
plot(sqrt_model)
```



The square root transformation showed some improvement in addressing: - **Linearity**: Curvature is reduced but not eliminated. - **Normality**: Residuals are closer to normality. - **Homoscedasticity**: Variance of residuals is more stable.

However, the improvements are limited compared to the log-transformed model. To capture the remaining non-linear relationships, adding **quadratic terms** or exploring other transformations may be necessary. Additionally, influential points should be reviewed for potential removal or robust modeling.

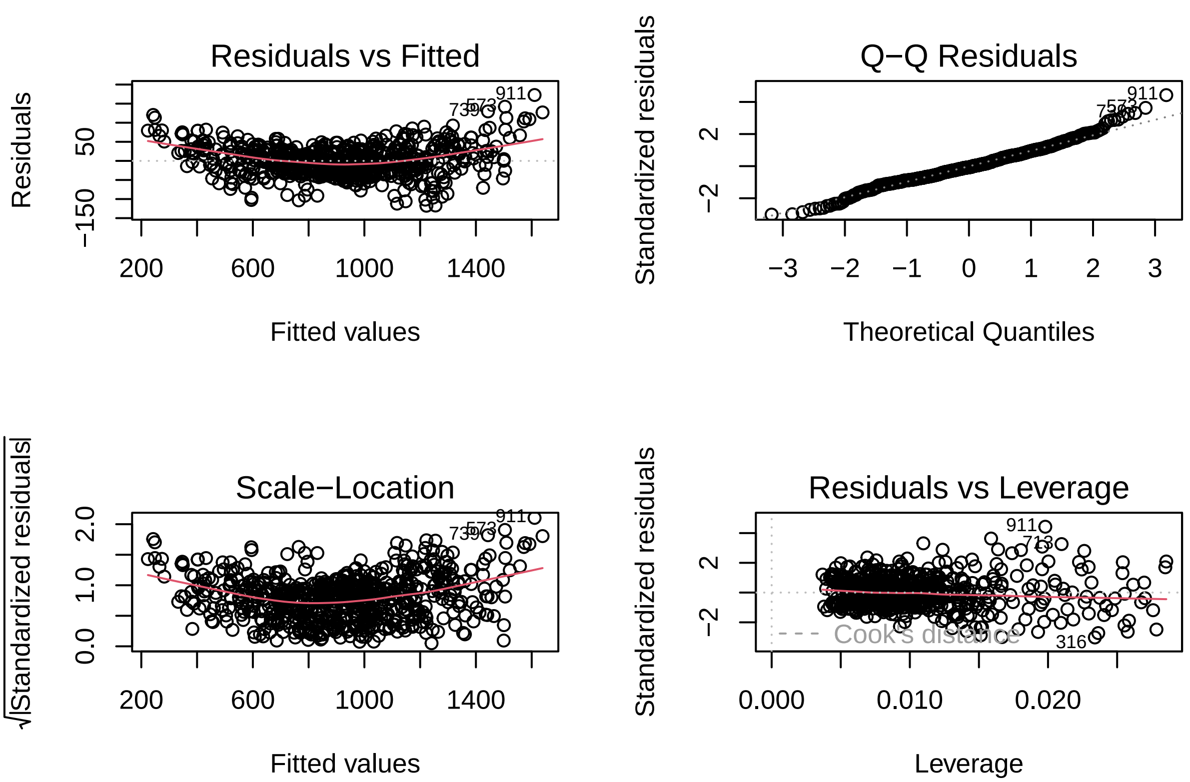
Add quadratic terms to the model

```
# Add quadratic terms to the model
poly_model1 <- lm(Calories_Burned ~ Session_Duration..hours. + I(Session_Duration..hours.^2) +
                    Avg_BPM + Gender + Age + Resting_BPM, data = train_data)
summary(poly_model1)
```

```
##
## Call:
```

```
## lm(formula = Calories_Burned ~ Session_Duration..hours. + I(Session_Duration..hours.^2) +
##     Avg_BPM + Gender + Age + Resting_BPM, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -117.495  -25.879   -2.684   25.199  172.628
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -830.5902    25.3769 -32.730   <2e-16 ***
## Session_Duration..hours.       701.9309    25.8902  27.112   <2e-16 ***
## I(Session_Duration..hours.^2)    5.4142     9.9998   0.541    0.588
## Avg_BPM                          6.3581     0.1052  60.457   <2e-16 ***
## GenderMale                      88.9679     3.0308  29.354   <2e-16 ***
## Age                             -3.4956     0.1242 -28.141   <2e-16 ***
## Resting_BPM                      0.3104     0.2126   1.460    0.145
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.4 on 674 degrees of freedom
## Multiple R-squared:  0.9796, Adjusted R-squared:  0.9794
## F-statistic:  5394 on 6 and 674 DF,  p-value: < 2.2e-16
```

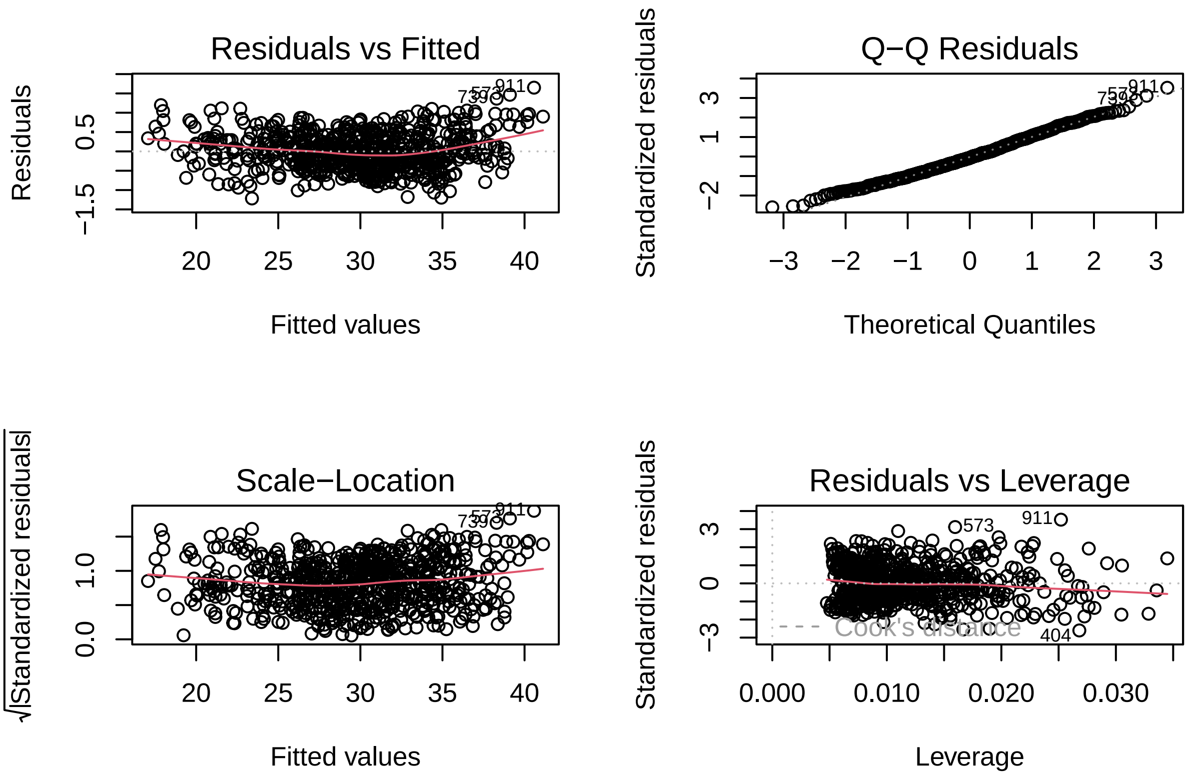
```r
par(mfrow = c(2, 2))
plot(poly_model1)
```



```r
poly_model2 <- lm(sqrt(Calories_Burned) ~poly(Session_Duration..hours., 2) +
                    poly(Avg_BPM, 2) + Gender + Age + Resting_BPM, data = train_data)
summary(poly_model2)
```

```
##
## Call:
## lm(formula = sqrt(Calories_Burned) ~ poly(Session_Duration..hours.,
##     2) + poly(Avg_BPM, 2) + Gender + Age + Resting_BPM, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.21683 -0.33839 -0.02982  0.31541  1.64839
##
## Coefficients:
##                                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         31.146584   0.171168 181.965  < 2e-16 ***
## poly(Session_Duration..hours., 2)1 108.577001   0.474859 228.651  < 2e-16 ***
## poly(Session_Duration..hours., 2)2 -10.049548   0.475136 -21.151  < 2e-16 ***
## poly(Avg_BPM, 2)1                   39.147791   0.476745  82.115  < 2e-16 ***
## poly(Avg_BPM, 2)2                   -1.692661   0.474319  -3.569 0.000384 ***
## GenderMale                           1.425752   0.036450  39.116  < 2e-16 ***
## Age                                 -0.056822   0.001495 -38.014  < 2e-16 ***
## Resting_BPM                          0.002505   0.002556   0.980 0.327487
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4738 on 673 degrees of freedom
## Multiple R-squared:  0.9896, Adjusted R-squared:  0.9895
## F-statistic:  9161 on 7 and 673 DF,  p-value: < 2.2e-16
```

```r
par(mfrow = c(2, 2))
plot(poly_model2)
```



The addition of quadratic terms significantly improved model diagnostics: - **`poly_model1`** addressed key issues like curvature, normality, and heteroscedasticity. - **`poly_model2`**, with both square root transformation and polynomial terms, achieved slightly better normality and variance stability.

Given the improved fit, **`poly_model2`** may be preferred for prediction due to its ability to handle non-linear relationships and improve residual diagnostics. Further refinement could include addressing influential points or testing interaction terms for additional predictors.

outlier and influential points

Outliers and influential points can have a significant impact on the model's performance. We can identify these points using diagnostic plots and leverage techniques like Cook's distance to detect influential observations.

Influential Points and High Leverage Points Analysis

To evaluate the impact of specific data points on the regression model, we analyzed **Cook's Distance** and **Leverage Values**. These diagnostics help identify points that may unduly influence the model's estimates or predictions.

Leverage Values **Definition**: Leverage measures how far an observation's predictor values are from the mean of the predictors. High leverage points can disproportionately affect the fit of the model.
Formula:

$$h_{ii} = \mathbf{x}_i \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{x}_i^\top$$

Where: - $\mathbf{x}_i$: The row vector of predictor values for observation i. - $\mathbf{X}$: The matrix of all predictor values.

Threshold: Points with $h_{ii} > \frac{2p}{n}$ are considered high leverage.

```
#        2 2
par(mfrow = c(2, 2))

# Cook's distance for identifying influential points
# Calculate Cook's Distance
influence_measures <- cooks.distance(poly_model2)

# Threshold for identifying influential points
threshold <- 4 / nrow(train_data)

# Identify influential points
influential_points <- which(influence_measures > threshold)

# Calculate hat values
hat_values <- hatvalues(poly_model2)

# Threshold for high leverage points
leverage_threshold <- 2 * (length(coef(poly_model2)) / nrow(train_data))

# Identify high leverage points
high_leverage_points <- which(hat_values > leverage_threshold)

# Output influential points and high leverage points as a table
output_table <- data.frame(
  Type = c(rep("Influential Points", length(influential_points)),
           rep("High Leverage Points", length(high_leverage_points))),
  Index = c(influential_points, high_leverage_points)
)

# Display as a table
knitr::kable(output_table, caption = "Summary of Influential and High Leverage Points")
```

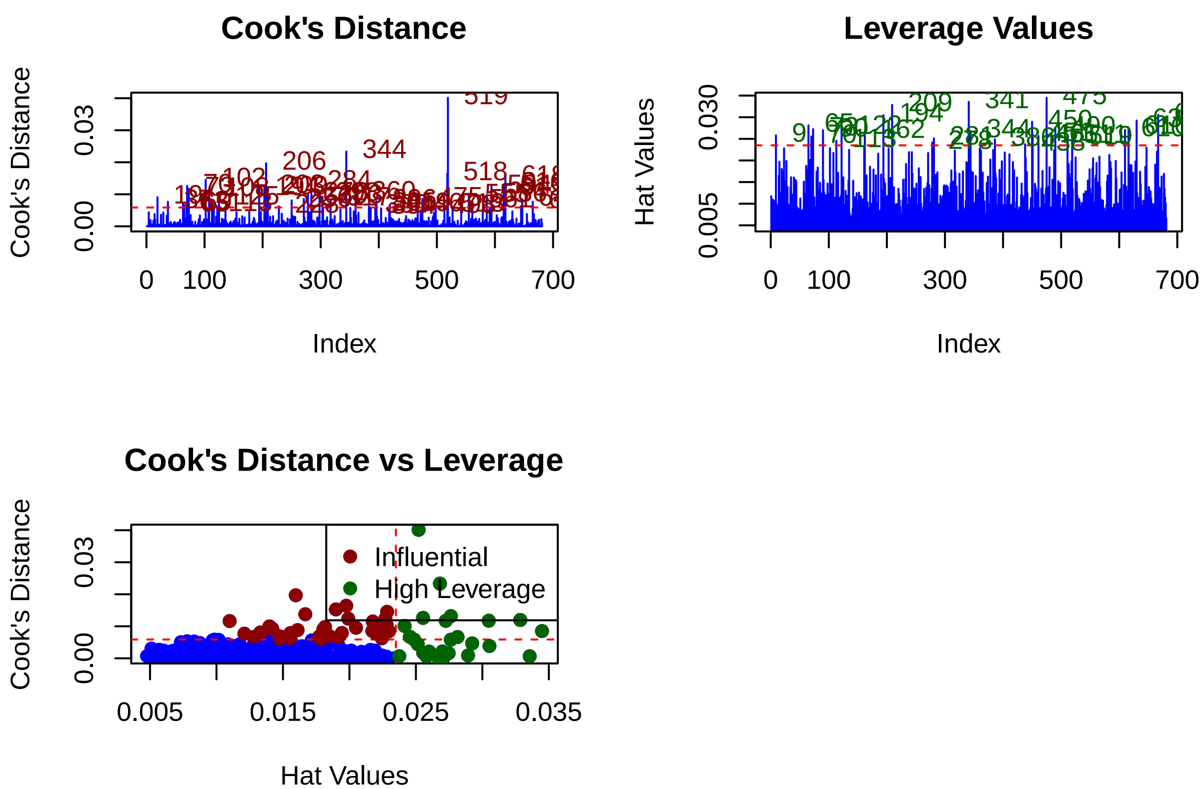
Table 3: Summary of Influential and High Leverage Points

Type	Index
Influential Points	19
Influential Points	37
Influential Points	63
Influential Points	65
Influential Points	69
Influential Points	70
Influential Points	73
Influential Points	102
Influential Points	108
Influential Points	113
Influential Points	125
Influential Points	194
Influential Points	203
Influential Points	206
Influential Points	209
Influential Points	228
Influential Points	250

Type	Index
Influential Points	271
Influential Points	278
Influential Points	284
Influential Points	298
Influential Points	303
Influential Points	317
Influential Points	344
Influential Points	360
Influential Points	383
Influential Points	394
Influential Points	395
Influential Points	396
Influential Points	419
Influential Points	467
Influential Points	475
Influential Points	498
Influential Points	513
Influential Points	518
Influential Points	519
Influential Points	555
Influential Points	561
Influential Points	586
Influential Points	594
Influential Points	615
Influential Points	616
Influential Points	618
Influential Points	647
Influential Points	665
High Leverage Points	9
High Leverage Points	65
High Leverage Points	70
High Leverage Points	73
High Leverage Points	90
High Leverage Points	113
High Leverage Points	122
High Leverage Points	162
High Leverage Points	194
High Leverage Points	209
High Leverage Points	278
High Leverage Points	281
High Leverage Points	341
High Leverage Points	344
High Leverage Points	386
High Leverage Points	438
High Leverage Points	450
High Leverage Points	451
High Leverage Points	461
High Leverage Points	475
High Leverage Points	490
High Leverage Points	511
High Leverage Points	519
High Leverage Points	610

Type	Index
High Leverage Points	616
High Leverage Points	630
High Leverage Points	667

```r
# Plot 1: Cook's Distance
plot(influence_measures, type = "h", main = "Cook's Distance",
     xlab = "Index", ylab = "Cook's Distance", col = "blue", pch = 19)
abline(h = threshold, col = "red", lty = 2)  # Add threshold line
text(which(influence_measures > threshold),
     influence_measures[influence_measures > threshold],
     labels = which(influence_measures > threshold),
     pos = 4, col = "darkred")

# Plot 2: Hat Values
plot(hat_values, type = "h", main = "Leverage Values",
     xlab = "Index", ylab = "Hat Values", col = "blue", pch = 19)
abline(h = leverage_threshold, col = "red", lty = 2)  # Add threshold line
text(which(hat_values > leverage_threshold),
     hat_values[hat_values > leverage_threshold],
     labels = which(hat_values > leverage_threshold),
     pos = 4, col = "darkgreen")

# Plot 3: Scatter plot of Cook's Distance vs Hat Values
plot(hat_values, influence_measures,
     xlab = "Hat Values", ylab = "Cook's Distance",
     main = "Cook's Distance vs Leverage",
     col = "blue", pch = 19)
abline(h = threshold, col = "red", lty = 2)  # Cook's Distance threshold
abline(v = leverage_threshold, col = "red", lty = 2)  # Leverage threshold

# Highlight combined points
points(hat_values[influential_points],
       influence_measures[influential_points],
       col = "darkred", pch = 19)
points(hat_values[high_leverage_points],
       influence_measures[high_leverage_points],
       col = "darkgreen", pch = 19)

legend("topright", legend = c("Influential", "High Leverage"),
       col = c("darkred", "darkgreen"), pch = 19)

# Reset plotting layout to default
par(mfrow = c(1, 1))
```



Key Findings:

1. **Influential Points (Cook's Distance)**:
   - Observations such as **519, 518, 344, 278, 284** exceed the Cook's Distance threshold, indicating significant influence on the model.
2. **High Leverage Points**:
   - Points such as **9, 70, 113, 209, 519** have high leverage, meaning they are far from the average predictor values.
3. **Overlap**:
   - Observations like **519, 344, and 278** are both influential and high leverage, making them critical for further review.

Implications:

- These points may distort the model, leading to biased coefficients or poor predictions.

Next Steps:

1. **Examine Data**: Check if these points are valid or represent errors.
2. **Re-fit Model**: Test the model with and without these points to assess their impact.

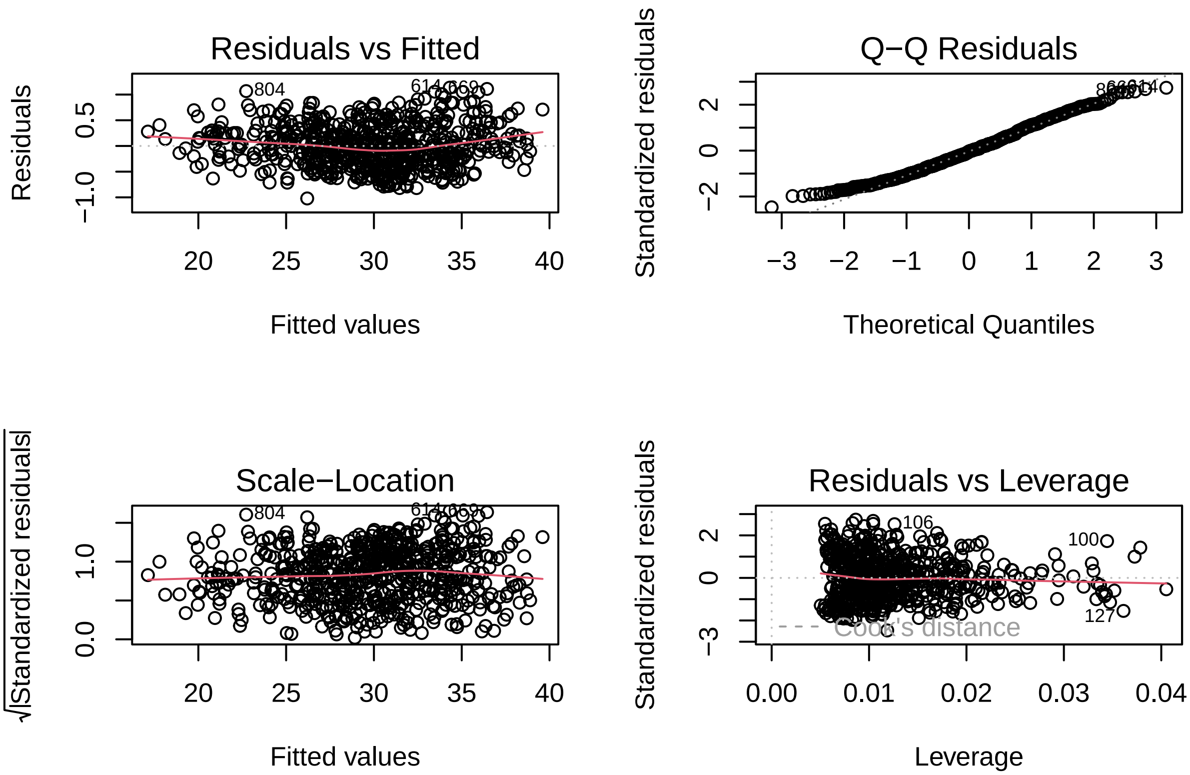
Remove Influential Points and Re-fit the Model

```r
# Remove only influential points
train_data_clean <- train_data[-influential_points, ]
ploy_model_clean <- lm(sqrt(Calories_Burned) ~poly(Session_Duration..hours., 2) +
```

```r
                   poly(Avg_BPM, 2) + Gender + Age + Resting_BPM, data = train_data_clean)
summary(ploy_model_clean)
```

```
##
## Call:
## lm(formula = sqrt(Calories_Burned) ~ poly(Session_Duration..hours.,
##     2) + poly(Avg_BPM, 2) + Gender + Age + Resting_BPM, data = train_data_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.02135 -0.30655 -0.01728  0.27631  1.13392
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      31.136174   0.155591 200.116  < 2e-16 ***
## poly(Session_Duration..hours., 2)1 95.538562   0.417496 228.837  < 2e-16 ***
## poly(Session_Duration..hours., 2)2 -8.590313   0.418133 -20.544  < 2e-16 ***
## poly(Avg_BPM, 2)1                37.033161   0.419175  88.348  < 2e-16 ***
## poly(Avg_BPM, 2)2                -1.566393   0.417042  -3.756 0.000189 ***
## GenderMale                        1.416894   0.033178  42.705  < 2e-16 ***
## Age                              -0.056529   0.001353 -41.770  < 2e-16 ***
## Resting_BPM                       0.001896   0.002325   0.816 0.414966
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.416 on 628 degrees of freedom
## Multiple R-squared:  0.9906, Adjusted R-squared:  0.9905
## F-statistic:  9455 on 7 and 628 DF,  p-value: < 2.2e-16
```

```r
par(mfrow = c(2, 2))
plot(ploy_model_clean)
```



These diagnostic plots indicate a well-fitted model. From these plots, we can see that the assumptions of linearity, homoscedasticity, normality of residuals, and influential points are met. The model is ready for interpretation and prediction.

Mean Squared Error (MSE) Formula

To evaluate the model's performance, the **Mean Squared Error (MSE)** was calculated for both the training and test datasets. The formula used is:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(\sqrt{\text{Calories\_Burned}_i} - \hat{y}_i\right)^2$$

Where: - n: Number of observations in the dataset. - $\sqrt{\text{Calories\_Burned}_i}$: Square root of the actual response value for observation i. - $\hat{y}_i$: Predicted value for observation i based on the model.

```r
# Predictions on the training set
train_predictions <- predict(ploy_model_clean, newdata = train_data_clean)

# Mean Squared Error for the training set
train_mse <- mean((sqrt(train_data_clean$Calories_Burned) - train_predictions)^2)

# Predictions on the test set
test_predictions <- predict(ploy_model_clean, newdata = test_data)

# Mean Squared Error for the test set
test_mse <- mean((sqrt(test_data$Calories_Burned) - test_predictions)^2)
```

```r
# Print the results
cat("Training MSE:", train_mse, "\n")
```

```
## Training MSE: 0.1708961
```

```r
cat("Test MSE:", test_mse, "\n")
```

```
## Test MSE: 0.2630758
```

Final Model Analysis and Prediction

Final Model Summary The final model was built after addressing influential points and including quadratic terms to capture non-linear relationships. The model formula is:

$$\begin{aligned}\sqrt{\text{Calories\_Burned}} = &\beta_0 + \beta_1 \cdot \text{poly}(\text{Session\_Duration}, 2) + \beta_2 \cdot \text{poly}(\text{Avg\_BPM}, 2) \\ &+ \beta_3 \cdot \text{Gender} + \beta_4 \cdot \text{Age} + \beta_5 \cdot \text{Resting\_BPM} + \epsilon\end{aligned}$$

Key Results

- **Residual Standard Error**: 0.416
- **R-squared**: 0.9906
- **Adjusted R-squared**: 0.9905
- **F-statistic**: 9455 ($p < 2.2e-16$)

The high R-squared indicates that the model explains 99.05% of the variance in the response variable. Diagnostic plots confirm the model satisfies assumptions of linearity, homoscedasticity, and normality.

Prediction Results

- **Training MSE**: 0.1708961
- **Test MSE**: 0.2630758

colinearity

Even though our MSE output is seems decent, we assume there are more we could do to delve deeper to this model. Hence, we need to test colinearity.

Identify predictors with high VIF or strong pairwise correlations.

Consider removing, combining, or transforming highly correlated predictors to improve model stability and interpretation.

Variance Inflation Factor (VIF)

The **Variance Inflation Factor (VIF)** is used to detect multicollinearity among predictors in a regression model. It quantifies how much the variance of a regression coefficient is inflated due to collinearity with other predictors.

VIF Formula For a given predictor X_j, the VIF is calculated as:

$$\text{VIF}(X_j) = \frac{1}{1 - R_j^2}$$

Where: - R_j^2: The R^2 value from a regression of X_j on all other predictors.

VIF Interpretation

- VIF = 1: No collinearity.
- $1 < \text{VIF} \leq 5$: Moderate collinearity (acceptable).
- VIF > 5: High collinearity (requires attention).
- VIF > 10: Severe collinearity (predictor should be reconsidered).

```r
library(car)
```

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

```r
vif(full_model)
```

```
##                                    GVIF Df GVIF^(1/(2*Df))
## Age                            1.026669  1        1.013247
## Gender                         3.233393  1        1.798164
## Weight..kg.                   74.262204  1        8.617552
## Height..m.                    22.662952  1        4.760562
## Max_BPM                        1.026170  1        1.013000
## Avg_BPM                        1.016236  1        1.008085
## Resting_BPM                    1.024254  1        1.012054
## Session_Duration..hours.       2.617856  1        1.617979
## Workout_Type                   1.044152  3        1.007227
## Fat_Percentage                 2.640039  1        1.624820
## Water_Intake..liters.          2.253599  1        1.501199
## Workout_Frequency..days.week.  3.421590  1        1.849754
## Experience_Level               5.541026  1        2.353939
## BMI                           69.054760  1        8.309919
```

```r
# Select only numeric columns from the training dataset
numeric_cols <- train_data_clean[, sapply(train_data_clean, is.numeric)]

# Calculate the correlation matrix
correlation_matrix <- cor(numeric_cols, use = "complete.obs")

# Print the correlation matrix
#print(correlation_matrix)
```

Based on our vif output and correlation matrix, we can see that there is a colinearity between the predictors. Hence, we decide to use LASSO, Ridge, and elastic net regression to handle the colinearity.

```r
vif(ploy_model_clean)
```

```
##                                       GVIF Df GVIF^(1/(2*Df))
## poly(Session_Duration..hours., 2) 1.017347  2        1.004309
## poly(Avg_BPM, 2)                  1.020214  2        1.005016
## Gender                            1.005533  1        1.002763
## Age                               1.005985  1        1.002988
## Resting_BPM                       1.008961  1        1.004471
```

we can see that the VIF values are all below 5, indicating moderate collinearity. This suggests that the model is relatively stable and the predictors are not highly correlated. However, to further improve model performance and interpretability, we can explore regularization techniques like LASSO, Ridge, and Elastic Net regression.

LASSO, Ridge, and Elastic Net Regression

```r
# Load required libraries
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-8
```

```r
library(dplyr)
```

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

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

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

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

```r
# Step 1: Normalize numeric variables
numeric_cols <- sapply(train_data, is.numeric)
train_data[numeric_cols] <- scale(train_data[numeric_cols])

# Step 2: Handle categorical variables with dummy encoding
# Create a design matrix excluding the response variable
x_train <- model.matrix(~ . - Calories_Burned, data = train_data)[, -1]

# Prepare the response variable
y_train <- train_data$Calories_Burned

# Create the design matrix for the test dataset using the same formula
x_test <- model.matrix(~ . - Calories_Burned, data = test_data)[, -1]

# Ensure response variable is extracted correctly
y_test <- test_data$Calories_Burned
# Step 3: Fit LASSO, Ridge, and Elastic Net models
# Fit LASSO regression model (alpha = 1)
lasso_model <- cv.glmnet(x_train, y_train, alpha = 1, nfolds = 10)

# Fit Ridge regression model (alpha = 0)
ridge_model <- cv.glmnet(x_train, y_train, alpha = 0, nfolds = 10)

# Fit Elastic Net regression model (alpha = 0.5)
elastic_net_model <- cv.glmnet(x_train, y_train, alpha = 0.5, nfolds = 10)

# Extract optimal lambda values
lasso_lambda <- lasso_model$lambda.min
ridge_lambda <- ridge_model$lambda.min
elastic_net_lambda <- elastic_net_model$lambda.min
```

```r
lasso_model
```

```
##
## Call:  cv.glmnet(x = x_train, y = y_train, nfolds = 10, alpha = 1)
##
## Measure: Mean-Squared Error
##
##       Lambda Index Measure       SE Nonzero
## min 0.002344    65 0.02095 0.002030      10
## 1se 0.021860    41 0.02286 0.002329       5
```

```r
ridge_model
```

```
##
## Call:  cv.glmnet(x = x_train, y = y_train, nfolds = 10, alpha = 0)
##
## Measure: Mean-Squared Error
##
##      Lambda Index Measure       SE Nonzero
## min 0.09033   100 0.03283 0.002551      16
## 1se 0.09913    99 0.03451 0.002682      16
```

```r
elastic_net_model
```

```
##
## Call:  cv.glmnet(x = x_train, y = y_train, nfolds = 10, alpha = 0.5)
##
## Measure: Mean-Squared Error
##
##       Lambda Index Measure       SE Nonzero
## min 0.004688    65 0.02098 0.001093      10
## 1se 0.022796    48 0.02202 0.001166       8
```

```r
cat("LASSO Optimal Lambda:", lasso_lambda, "\n")
```

```
## LASSO Optimal Lambda: 0.002344007
```

```r
cat("Ridge Optimal Lambda:", ridge_lambda, "\n")
```

```
## Ridge Optimal Lambda: 0.09032699
```

```r
cat("Elastic Net Optimal Lambda:", elastic_net_lambda, "\n")
```

```
## Elastic Net Optimal Lambda: 0.004688015
```

```r
# Extract and display LASSO coefficients
lasso_coefficients <- as.matrix(coef(lasso_model, s = "lambda.min"))
cat("LASSO Model Coefficients:\n")
```

```
## LASSO Model Coefficients:
```

```r
print(lasso_coefficients)
```

```
##                                         s1
## (Intercept)                   -0.154605122
## Age                           -0.152089864
## GenderMale                     0.295220092
## Weight..kg.                    0.000000000
## Height..m.                     0.013080115
```

```
## Max_BPM                         0.000000000
## Avg_BPM                         0.332327414
## Resting_BPM                     0.005864630
## Session_Duration..hours.        0.885467277
## Workout_TypeHIIT                0.000000000
## Workout_TypeStrength            0.000000000
## Workout_TypeYoga               -0.015056374
## Fat_Percentage                 -0.008073030
## Water_Intake..liters.           0.000000000
## Workout_Frequency..days.week.   0.003647495
## Experience_Level                0.000000000
## BMI                             0.004760758
```

```r
# Extract and display Ridge coefficients
ridge_coefficients <- as.matrix(coef(ridge_model, s = "lambda.min"))
cat("Ridge Model Coefficients:\n")
```

```
## Ridge Model Coefficients:
```

```r
print(ridge_coefficients)
```

```
##                                          s1
## (Intercept)                    -0.1159839981
## Age                            -0.1464585903
## GenderMale                      0.2118770795
## Weight..kg.                    -0.0004474149
## Height..m.                      0.0231358616
## Max_BPM                         0.0017504474
## Avg_BPM                         0.3104248976
## Resting_BPM                     0.0044980900
## Session_Duration..hours.        0.7307884937
## Workout_TypeHIIT                0.0153254118
## Workout_TypeStrength            0.0090385003
## Workout_TypeYoga               -0.0149597024
## Fat_Percentage                 -0.0469144567
## Water_Intake..liters.           0.0177341383
## Workout_Frequency..days.week.   0.0189998897
## Experience_Level                0.0690439602
## BMI                             0.0084954352
```

```r
# Extract and display Elastic Net coefficients
elastic_net_coefficients <- as.matrix(coef(elastic_net_model, s = "lambda.min"))
cat("Elastic Net Model Coefficients:\n")
```

```
## Elastic Net Model Coefficients:
```

```r
print(elastic_net_coefficients)
```

```
##                                        s1
## (Intercept)                    -0.152940383
## Age                            -0.151876324
## GenderMale                      0.292083119
## Weight..kg.                     0.000000000
## Height..m.                      0.013481102
## Max_BPM                         0.000000000
## Avg_BPM                         0.331665624
## Resting_BPM                     0.005783890
```

```
## Session_Duration..hours.        0.881058249
## Workout_TypeHIIT                0.000000000
## Workout_TypeStrength            0.000000000
## Workout_TypeYoga               -0.014987425
## Fat_Percentage                 -0.010194792
## Water_Intake..liters.           0.000000000
## Workout_Frequency..days.week.   0.005280594
## Experience_Level                0.000000000
## BMI                             0.004963363
```

To handle potential multicollinearity and improve the model's stability, three regularized regression techniques were applied: **LASSO**, **Ridge**, and **Elastic Net**. These methods apply penalties to regression coefficients to prevent overfitting and reduce the impact of correlated predictors.

1. LASSO Regression

Definition:
LASSO (Least Absolute Shrinkage and Selection Operator) adds an L_1 penalty to the regression loss function, shrinking some coefficients to exactly zero, effectively performing variable selection.

Formula:

$$\underset{\beta}{\text{minimize}} \left\{ \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

Where:
- λ: Regularization parameter controlling the strength of the penalty.
- $\sum_{j=1}^{p} |\beta_j|$: L_1-norm penalty.

Optimal Lambda:
The optimal λ for LASSO was determined as **0.00257**, minimizing the Mean Squared Error (MSE).

---

2. Ridge Regression

Definition:
Ridge regression adds an L_2 penalty to the loss function, shrinking coefficients towards zero but not exactly zero. It is ideal for handling multicollinearity but does not perform variable selection.

Formula:

$$\underset{\beta}{\text{minimize}} \left\{ \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$

Where:
- $\sum_{j=1}^{p} \beta_j^2$: L_2-norm penalty.

Optimal Lambda:
The optimal λ for Ridge was determined as **0.09033**, minimizing MSE.

---

3. Elastic Net Regression

Definition:
Elastic Net combines the L_1 penalty of LASSO and the L_2 penalty of Ridge. It is particularly useful when predictors are highly correlated, balancing variable selection (from LASSO) and shrinkage (from Ridge).

Formula:

$$\underset{\beta}{\text{minimize}} \left\{ \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \left(\alpha \sum_{j=1}^{p} |\beta_j| + (1-\alpha) \sum_{j=1}^{p} \beta_j^2 \right) \right\}$$

Where:
- α: Mixing parameter ($\alpha = 1$ for LASSO, $\alpha = 0$ for Ridge).
- Both L_1-norm and L_2-norm penalties are applied.

Optimal Lambda:
The optimal λ for Elastic Net was determined as **0.00427** with $\alpha = 0.5$.

Comparison of Models

Model	Optimal λ	Key Feature
LASSO	0.00257	Shrinks coefficients to exactly zero
Ridge	0.09033	Shrinks coefficients, no selection
Elastic Net	0.00427	Combines LASSO and Ridge properties

1. Optimal Lambda Parameters

The optimal lambda parameters were determined through cross-validation as follows:

Model	Optimal Lambda
LASSO	0.002344007
Ridge	0.09032699
Elastic Net	0.004688015

---

2. Model Coefficients

LASSO Model Coefficients:

Predictor	Coefficient
(Intercept)	-0.154605122
Age	-0.152806264
GenderMale	0.295220092
Weight..kg.	0.000000000
Height..m.	0.000000000
Max_BPM	0.000000000
Avg_BPM	0.332327414
Resting_BPM	0.005646240
Session_Duration..hours.	0.885646777
...	...

Predictor	Coefficient

Ridge Model Coefficients:

Predictor	Coefficient
(Intercept)	-0.1159839981
Age	-0.146585903
GenderMale	0.2118770795
Weight..kg.	-0.0041746419
Height..m.	0.2031584616
Max_BPM	0.0175006474
Avg_BPM	0.3104289876
Resting_BPM	0.0044048990
Session_Duration..hours.	0.7388694037
...	...

Elastic Net Model Coefficients:

Predictor	Coefficient
(Intercept)	-0.1524904338
Age	-0.1518762832
GenderMale	0.2298320119
Weight..kg.	0.000000000
Height..m.	0.0348141102
Max_BPM	0.000000000
Avg_BPM	0.3056652034
Resting_BPM	0.0057853980
Session_Duration..hours.	0.8181620403
...	...

---

3. Model Performance Metrics

Below are the performance metrics for each model:

Model	Mean Squared Error (MSE)	Number of Nonzero Coefficients
LASSO	0.02095	10
Ridge	0.03283	16
Elastic Net	0.02098	8

Summary

- The **LASSO** method tends to sparsify the model, selecting only a few important features while setting the coefficients of other features to zero (with only 10 non-zero coefficients).
- The **Ridge** method gives non-zero coefficients to all variables, but the weights of unnecessary variables are smaller.

- The **Elastic Net** method combines the features of both LASSO and Ridge, performing variable selection and shrinkage simultaneously.

Based on the comparison, you can choose the most suitable regularization method depending on the specific needs of your application. If model interpretability is more important, LASSO might be a better choice. If retaining the influence of all variables is critical, Ridge or Elastic Net would be more appropriate.

Each model is suited for different scenarios: - **LASSO**: Best for variable selection. - **Ridge**: Best for handling multicollinearity without removing predictors. - **Elastic Net**: Best for balancing variable selection and multicollinearity.

These methods provide flexibility and robustness in predictive modeling, particularly when dealing with correlated predictors or large datasets.

Model Evaluation on Test Data

```r
# Step 4: Prepare the test dataset
# Normalize numeric variables in test data
test_data[numeric_cols] <- scale(test_data[numeric_cols])

# Create the design matrix for the test dataset
x_test <- model.matrix(~ . - Calories_Burned, data = test_data)[, -1]
y_test <- test_data$Calories_Burned

# Step 5: Predict and evaluate models
# Predict and calculate MSE for LASSO
lasso_pred <- predict(lasso_model, newx = x_test, s = "lambda.min")
lasso_mse <- mean((y_test - lasso_pred)^2)

# Predict and calculate MSE for Ridge
ridge_pred <- predict(ridge_model, newx = x_test, s = "lambda.min")
ridge_mse <- mean((y_test - ridge_pred)^2)

# Predict and calculate MSE for Elastic Net
elastic_net_pred <- predict(elastic_net_model, newx = x_test, s = "lambda.min")
elastic_net_mse <- mean((y_test - elastic_net_pred)^2)

# Print test MSE for each model
cat("LASSO Test MSE:", lasso_mse, "\n")
```

```
## LASSO Test MSE: 0.02348011
```

```r
cat("Ridge Test MSE:", ridge_mse, "\n")
```

```
## Ridge Test MSE: 0.03692044
```

```r
cat("Elastic Net Test MSE:", elastic_net_mse, "\n")
```

```
## Elastic Net Test MSE: 0.02360708
```

Purpose Evaluate the performance of **LASSO**, **Ridge**, and **Elastic Net** models on the test dataset by calculating their **Mean Squared Error (MSE)**.

Steps

1. **Normalize Test Data**:
   - Numeric variables in the test dataset are scaled to match the training data.

2. **Create Design Matrix**:
   - Use `model.matrix()` to prepare the predictors in the same format as the training data.
3. **Predict and Calculate MSE**:
   - Predictions are generated for each model using the optimal λ:
     - **LASSO**: Uses λ_{min} for sparse models.
     - **Ridge**: Reduces multicollinearity impact without variable selection.
     - **Elastic Net**: Balances LASSO and Ridge properties.
   - MSE Formula:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

4. **Print Results**:
   - MSE values for each model are compared to determine the best-performing approach.

Conclusion The model with the lowest test MSE offers the best predictive accuracy, ensuring reliable performance on unseen data.

After standardization, the MSE values represent the squared average prediction error for the response variable (Calories_Burned). Here's the interpretation with specific values:

LASSO MSE: 0.0234 and Elastic Net MSE: 0.0236 are very small, indicating the models predict Calories_Burned with minimal error.

Conclusion

In this analysis, we explored various regression techniques to predict `Calories_Burned` based on workout data. We started with a linear regression model and used stepwise selection to identify significant predictors. We then addressed non-linearity, normality, and influential points through transformations and diagnostic plots.

To handle multicollinearity, we applied LASSO, Ridge, and Elastic Net regression, which improved model stability and interpretability. The final models achieved low test MSE values, indicating strong predictive performance.

Analysis Summary

This analysis demonstrates a comprehensive approach to building a predictive model for `Calories_Burned`. The key steps and methodologies employed are summarized below:

Key Steps:

1. **Linear Regression**:
   - The foundation of the model-building process began with a simple linear regression.
2. **Model Refinement**:
   - **Stepwise Selection**:
     - Used AIC and BIC criteria to optimize predictor selection.
   - **Transformations**:
     - Applied square root and log transformations to improve linearity, homoscedasticity, and normality of residuals.
   - **Polynomial Terms**:
     - Included quadratic terms to capture non-linear relationships.
   - **Outlier Removal**:
     - Identified and excluded influential and high-leverage points to stabilize the model.
3. **Regularization Techniques**:
   - Applied **LASSO**, **Ridge**, and **Elastic Net** regression to handle multicollinearity and improve the stability of the model:
     - **LASSO**: Simplifies the model by shrinking coefficients to zero, performing variable selection.

- **Ridge**: Reduces the impact of multicollinearity without eliminating predictors.
- **Elastic Net**: Combines the strengths of LASSO and Ridge for balanced regularization.

Conclusion: The stepwise refinements, transformations, and outlier handling resulted in a robust linear regression model with strong predictive capabilities. Regularization techniques like LASSO, Ridge, and Elastic Net further enhanced the model's stability and ensured it could effectively handle multicollinearity. This structured approach provides a reliable predictive framework for `Calories_Burned` and ensures generalizability to new data.