Statistical Model Building Using Fitbit Data in R Programming

```r
# Installing and Loading Required Packages
install.packages("tidyverse")
library(tidyverse)
library(dplyr)
```

```r
# Reading FitBit.csv File from the Directory
data = read.csv("FitBit Data.csv")
head(data,5)
```

	Sleeping	HeartRate	Glucose	Cholesterol	Run	BMI	Weight	Height	HealthIndex
	<int>	<int>	<int>	<int>	<int>	<dbl>	<int>	<dbl>	<int>
1	8	72	77	77	800	27.76	85	1.75	10
2	9	72	200	78	451	32.74	87	1.63	10
3	10	70	100	79	422	26.99	78	1.70	10
4	8	70	110	78	400	28.41	89	1.77	10
5	7	72	45	79	480	29.76	85	1.69	10

5 rows | 1-10 of 10 columns

```r
# Getting the Structure of the Data frame called data
str(data)
```

```
'data.frame':   50 obs. of  10 variables:
 $ Sleeping   : int  8 9 10 8 7 5 9 6 10 10 ...
 $ HeartRate  : int  72 72 70 70 72 89 75 85 72 78 ...
 $ Glucose    : int  77 200 100 110 45 78 88 89 250 141 ...
 $ Cholesterol: int  77 78 79 78 79 79 80 85 84 86 ...
 $ Run        : int  800 451 422 400 480 1000 1254 1452 800 899 ...
 $ BMI        : num  27.8 32.7 27 28.4 29.8 ...
 $ Weight     : int  85 87 78 89 85 85 87 89 110 85 ...
 $ Height     : num  1.75 1.63 1.7 1.77 1.69 1.52 1.5 1.4 1.75 1.75 ...
 $ HealthIndex: int  10 10 10 10 10 5 6 4 5 10 ...
 $ Condition  : chr  "Good" "Good" "Good" "Good" ...
```

```
# Converting the Condition Column to Factor type
data$Condition = factor(data$Condition)
str(data)
```

```
'data.frame':   50 obs. of  10 variables:
 $ Sleeping   : int  8 9 10 8 7 5 9 6 10 10 ...
 $ HeartRate  : int  72 72 70 70 72 89 75 85 72 78 ...
 $ Glucose    : int  77 200 100 110 45 78 88 89 250 141 ...
 $ Cholesterol: int  77 78 79 78 79 79 80 85 84 86 ...
 $ Run        : int  800 451 422 400 480 1000 1254 1452 800 899 ...
 $ BMI        : num  27.8 32.7 27 28.4 29.8 ...
 $ Weight     : int  85 87 78 89 85 85 87 89 110 85 ...
 $ Height     : num  1.75 1.63 1.7 1.77 1.69 1.52 1.5 1.4 1.75 1.75 ...
 $ HealthIndex: int  10 10 10 10 10 5 6 4 5 10 ...
 $ Condition  : Factor w/ 2 levels "Bad","Good": 2 2 2 2 2 1 1 1 1 2 ...
```

```
# Deselecting the Condition Column and Store it in a data frame
data2 = data %>% select(-Condition)
head(data2,5)
```

	Sleeping	HeartRate	Glucose	Cholesterol	Run	BMI	Weight	Height	HealthIndex
	<int>	<int>	<int>	<int>	<int>	<dbl>	<int>	<dbl>	<int>
1	8	72	77	77	800	27.76	85	1.75	10
2	9	72	200	78	451	32.74	87	1.63	10
3	10	70	100	79	422	26.99	78	1.70	10
4	8	70	110	78	400	28.41	89	1.77	10
5	7	72	45	79	480	29.76	85	1.69	10

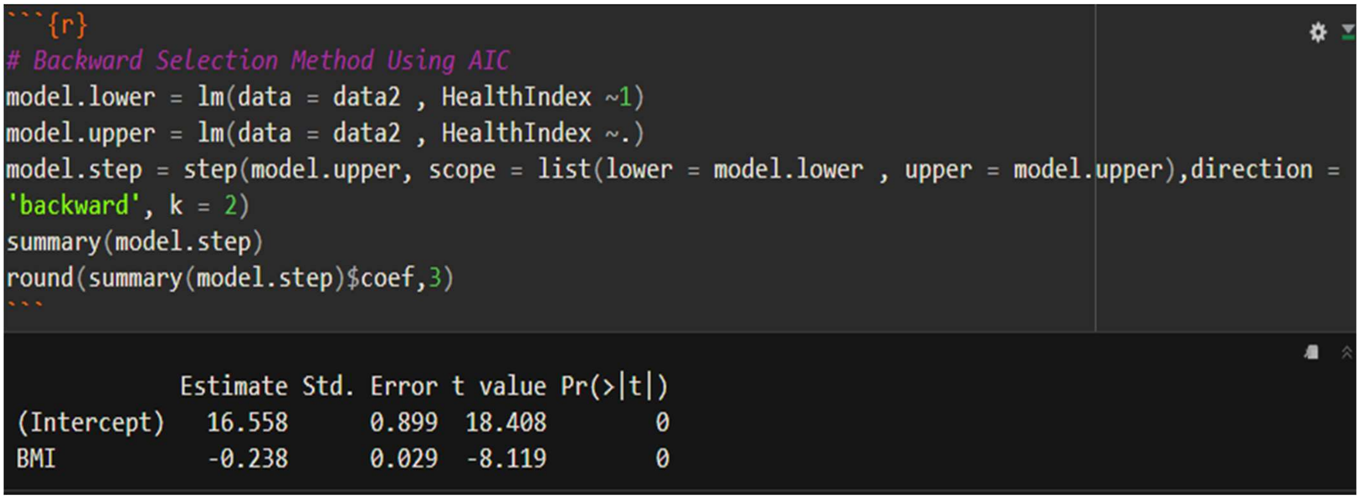
5 rows

```
# Building a Multiple Linear Regression Model using only Continuous predictor
s. Here we consider the HealthIndex as Response variable and the other factor
s as Predictor variables.
model <- lm(HealthIndex ~. , data = data2)
summary(model)
```

```

Call:
lm(formula = HealthIndex ~ ., data = data2)
```

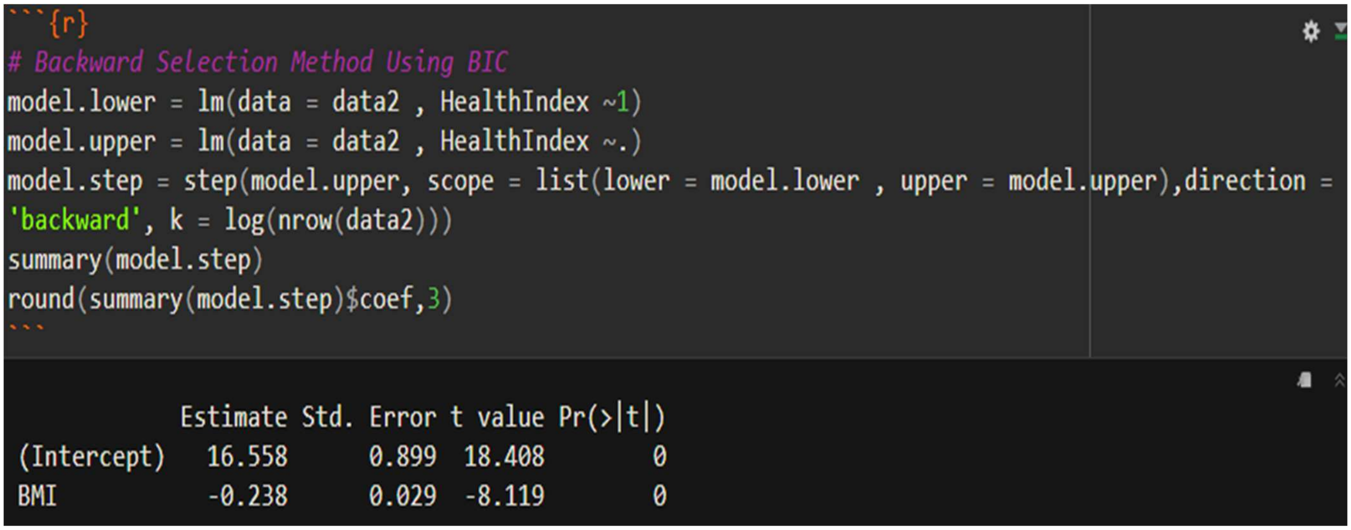
```
Residuals:
     Min       1Q   Median       3Q      Max
-3.13513 -0.39153  0.01525  0.44954  1.78686

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.054e+01  1.219e+01   3.326  0.00186 **
Sleeping     9.017e-03  8.010e-02   0.113  0.91092
HeartRate   -1.985e-02  1.009e-02  -1.968  0.05582 .
Glucose     -1.389e-03  3.882e-03  -0.358  0.72237
Cholesterol -8.544e-03  4.823e-03  -1.772  0.08386 .
Run          2.175e-04  2.158e-04   1.008  0.31958
BMI         -5.748e-01  1.715e-01  -3.353  0.00173 **
Weight       1.152e-01  6.634e-02   1.737  0.08995 .
Height      -1.268e+01  7.239e+00  -1.752  0.08722 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.022 on 41 degrees of freedom
Multiple R-squared:  0.6746,    Adjusted R-squared:  0.6111
F-statistic: 10.63 on 8 and 41 DF,  p-value: 6.288e-08
```

Result of The Above Model: As the result is shown the Predictor BMI only has a strong linear relationship with the Response as its P value is less than the critical value we consider. The model is approximately 61 % accurate. The rest 39 % is not explained by the model. Now we will try to increase the Model Accuracy by conducting Backward Selection Method.



```{r}
# Backward Selection Method Using AIC
model.lower = lm(data = data2 , HealthIndex ~1)
model.upper = lm(data = data2 , HealthIndex ~.)
model.step = step(model.upper, scope = list(lower = model.lower , upper = model.upper),direction =
'backward', k = 2)
summary(model.step)
round(summary(model.step)$coef,3)
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   16.558      0.899  18.408        0
BMI           -0.238      0.029  -8.119        0
```



~~~
```{r}
# Backward Selection Method Using BIC
model.lower = lm(data = data2 , HealthIndex ~1)
model.upper = lm(data = data2 , HealthIndex ~.)
model.step = step(model.upper, scope = list(lower = model.lower , upper = model.upper),direction =
'backward', k = log(nrow(data2)))
summary(model.step)
round(summary(model.step)$coef,3)
```
~~~

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   16.558      0.899  18.408        0
BMI           -0.238      0.029  -8.119        0
```

Interpretation of Backward Selection Using AIC and BIC: Both the AIC and BIC method of Backward Selection gives the same result. The result is obtained at the very first time while building the Multiple Linear Regression Model. This Backward Selection method is a way to increase the accuracy of the model and simplify the model by reducing number of predictors.