Project report

STATISTICAL ANALYSIS ON FISH WEIGHT IN RELATION TO ITS CHARACTERISTICS

Kulpunai Kurstanbek Kyzy

SP Jain School of Global Management
Data science 2021

Table of Contents

Introduction

Fish is a paraphyletic group (according to modern cladistic filtration) of aquatic vertebrates. Fish live in both salty and fresh waters - from the shores of ocean depressions to mountain streams. Fish play an important role in most aquatic ecosystems as part of the food chain. Many species of fish are used by humans for food and therefore are of great commercial importance.

This study focuses on the impact of each condition factor on fish weight. The purpose of the study is to predict the weight of different fish species based on its characteristics like vertical and diagonal length, height, and width. We will use Linear Regression Method to see whether the weight of the fish related to their characteristics.

With Linear Regression Method, we will use fish dataset from Kaggle to identify the relationship between Weight with other numerical variables. We also try to see whether the weight of the fish can be predicted based on historical data.

There are 6 columns and 159 rows. Here are the descriptions about dataset:

1. Weight: Weight of the fish, in grams
2. Length1: Vertical length of the fish, in centimeters. Renamed to Vertical
3. Length2: Diagonal length of the fish, in centimeters. Renamed to Diagonal
4. Length3: Cross length of the fish, in centimeters. Renamed to Cross
5. Height: Height of the fish, in centimeters
6. Width: Diagonal width of the fish, in centimeters

Weight is the dependant variable to be studied based on the input variables

Note: All the figure henceforth shown in the report will be an output from R-code

We will use the following libraries for our analysis:

```
library(tidyverse)
library(caret)
library(plotly)
library(data.table)
library(GGally)
library(car)
library(scales)
library(lmtest)
library(ggplot2)
library(performance)
library(MLmetrics)
library(rmdformats)
```

Data Wrangling

We will exclude Species variable as we do not need it now. We will use only the numerical variable. Let's have a look at data types of variables.

```
> #check data type
> glimpse(fish)
Rows: 159
Columns: 6
$ Weight  <dbl> 242, 290, 340, 363, 430, 450, 500, 390, 450, 500, 475, 500, 500…
$ Length1 <dbl> 23.2, 24.0, 23.9, 26.3, 26.5, 26.8, 26.8, 27.6, 27.6, 28.5, 28.…
$ Length2 <dbl> 25.4, 26.3, 26.5, 29.0, 29.0, 29.7, 29.7, 30.0, 30.0, 30.7, 31.…
$ Length3 <dbl> 30.0, 31.2, 31.1, 33.5, 34.0, 34.7, 34.5, 35.0, 35.1, 36.2, 36.…
$ Height  <dbl> 11.5200, 12.4800, 12.3778, 12.7300, 12.4440, 13.6024, 14.1795, …
$ Width   <dbl> 4.0200, 4.3056, 4.6961, 4.4555, 5.1340, 4.9274, 5.2785, 4.6900,…
>
```

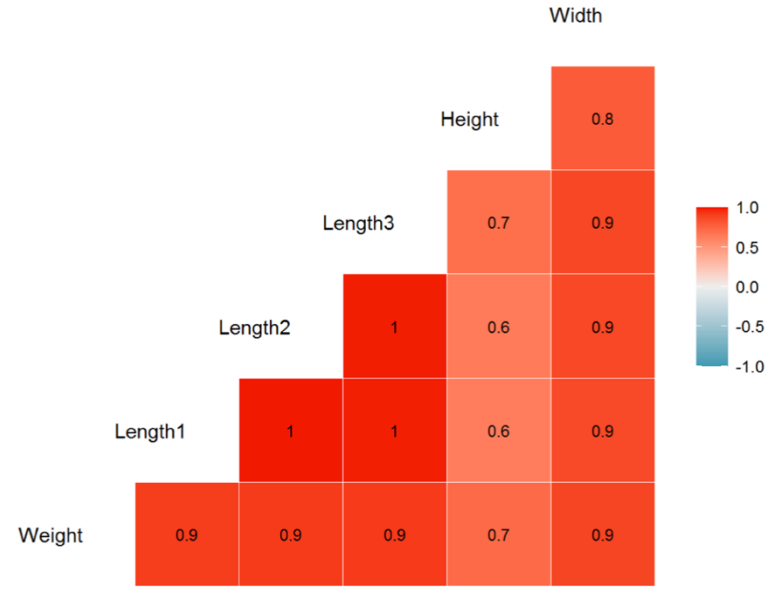
Null value test was performed on the dataset. No missing values were present in the dataset.

Exploratory Data Analysis

Exploratory data analysis is a way to better understand your data which helps in further steps. And data visualization makes the exploratory data analysis process easier.

In order to analyse our features more carefully, we look at the correlation of various features of the fish species.

We will use Pearson correlation with ggcorr function from GGally Library.



From the graphic shown above we can see how each of the numerical variables have a strong correlation with Weight as our target variable.

Cross Validation

Before the section of model building, we need to split our dataset into train and test dataset for more accurate model. We will use train dataset to train our model, while the test dataset will be used as a comparison whether our model can predict correctly new data that has not been used.

We will split the train and test dataset with 70 : 30 ratio.

```
#cross validation
set.seed(100)
index <- sample(nrow(fish), nrow(fish)*0.7)

fish_train <- fish[index,]
fish_test <- fish[-index,]
```

Modelling

At this phase, we will try to create a model with Linear Regression Method with Weight as our target variable and Width as our predictor variable.

```
> #modelling
> set.seed(100)
> model_init <- lm(formula = Weight ~ Width, data = fish_train)
> summary(model_init)

Call:
lm(formula = Weight ~ Width, data = fish_train)

Residuals:
    Min      1Q  Median      3Q     Max
-250.02 -110.64  -37.03   62.77  887.31

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -423.838     46.690  -9.078 5.41e-15 ***
Width        184.982      9.954  18.584  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 171.7 on 109 degrees of freedom
Multiple R-squared:  0.7601,    Adjusted R-squared:  0.7579
F-statistic: 345.4 on 1 and 109 DF,  p-value: < 2.2e-16
```

From the summary, we can conclude that Width as our predictor variable have p-value below 0.05, this means Width have a significant effect toward Weight as our target variable.For simple interpretation of the coefficient, every increased 1 unit point in Width, it will contribute to 184.982 increase in Weight.

We also need to check the Multiple R-Square of the model, we can see that our Multiple R-Square is around 0.7601 or 76.01%. This means that the model can explain 76.01% of variance of our target variable.

Now we will try to add more variables to our model. To do this we will use Step-Wise Regression Method. This method will create an optimum formula for our model in terms of lowest AIC.The lower the result of AIC, the less the undetected observation value will be.

First we will create two different models. model_null is the model without any predictor variable, and model_all with all variables included to it.

```
> model_null <- lm(formula = Weight ~ 1, data = fish_train)
> summary(model_null)

Call:
lm(formula = Weight ~ 1, data = fish_train)

Residuals:
   Min     1Q Median     3Q    Max
-389.2 -269.2 -119.2  260.8 1210.8

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   389.24      33.13   11.75   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 349 on 110 degrees of freedom
```

```
> model_all <- lm(formula = Weight ~ ., data = fish_train)
> summary(model_all)

Call:
lm(formula = Weight ~ ., data = fish_train)

Residuals:
    Min      1Q  Median      3Q     Max
-251.54  -62.93  -20.95   44.49  437.95

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -487.479     35.933 -13.566  < 2e-16 ***
Length1       49.752     47.601   1.045 0.298327
Length2       16.237     50.058   0.324 0.746314
Length3      -38.222     20.118  -1.900 0.060196 .
Height        34.218      9.926   3.447 0.000815 ***
Width          1.121     23.284   0.048 0.961689
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 123.3 on 105 degrees of freedom
Multiple R-squared:  0.881,     Adjusted R-squared:  0.8753
F-statistic: 155.4 on 5 and 105 DF,  p-value: < 2.2e-16
```

Based on model_null, we can see without any predictors included, it will create the mean value of the target variable with the value of 389.24. For our model_all we can conclude that when all variables were put into the model, the only variable with p-value < 0.05 is the Height variable.

Now we will use Stepwise Regression Method to choose our best model to be analyzed.

```
# Create formula using Step-Wise Regression
step(object = model_all, direction = "backward", trace = F)

step(object = model_null, direction = "forward", scope = list(lower = model_null, upper = model_all), trace = F)

step(object = model_null, direction = "both", scope = list(lower = model_null, upper = model_all), trace = F)

# Create the model

model_step_back <- lm(formula = Weight ~ Length1 + Length3 + Height, data = fish_train)
model_step_forw <- lm(formula = Weight ~ Length3 + Width + Height + Length1, data = fish_train)
model_step_both <- lm(formula = Weight ~ Length3 + Height + Length1, data = fish_train)
```

Let's compare the models we have just created, so we can choose the best one based on comparison performance:

```
> # Compare these models with `model_all`
> compare_performance(model_all, model_step_back, model_step_forw, model_step_both)
# Comparison of Model Performance Indices

Name            | Model |      AIC | AIC weights |      BIC | BIC weights |    R2 | R2 (adj.) |    RMSE |   Sigma
-----------------------------------------------------------------------------------------------------------------
model_all       |    lm | 1391.594 |       0.057 | 1410.560 |       0.005 | 0.881 |     0.875 | 119.874 | 123.251
model_step_back |    lm | 1387.728 |       0.397 | 1401.275 |       0.475 | 0.881 |     0.877 | 119.946 | 122.168
model_step_forw |    lm | 1389.705 |       0.148 | 1405.962 |       0.046 | 0.881 |     0.876 | 119.934 | 122.730
model_step_both |    lm | 1387.728 |       0.397 | 1401.275 |       0.475 | 0.881 |     0.877 | 119.946 | 122.168
```

From the comparison between 4 models it is clear that models with the lowest AIC value are model_step_back and model_step_both. Both of the models have an AIC value of 1387.73. Both of the models are also have an Adjusted R-Square around 0.88 or 88%. This means that both of the model that we create before can explain 88% of variance of our target variable.

It doesn't matter which model to choose, since the results we need from these two models are the same. Therefore, we will continue with model_step_both for further analysis.

Evaluation:

The most common metric for evaluating linear regression model performance is called root mean squared error, or RMSE. The basic idea is to measure how bad/good the model's predictions are when compared to actual observed values.

```
> #evaluation
> fish_pred <- predict(object = model_step_both, newdata = fish_test, level = 0.95)
>
> # RMSE of train dataset
> RMSE(y_pred = model_step_both$fitted.values, y_true = fish_train$Weight)
[1] 119.9461
> # RMSE of test dataset
> RMSE(y_pred = fish_pred, y_true = fish_test$Weight)
[1] 124.8925
>
```

We can see from the RMSE that the model that have been trained by our training dataset are good enough to predict testing dataset. It is shown by the error of the train dataset which is is lower than the value test dataset.

Assumption Checking

Multiple linear regression analysis makes several key assumptions:

- Normality – Multiple regression assumes that the residuals are normally distributed
- Homoscedasticity- This assumption states that the variance of error terms are similar across the values of the independent variables. A plot of standardized residuals versus predicted values can show whether points are equally distributed across all values of the independent variables.
- No Multicollinearity - Multiple regression assumes that the independent variables are not highly correlated with each other. This assumption is tested using Variance Inflation Factor (VIF) values.

Normality test:

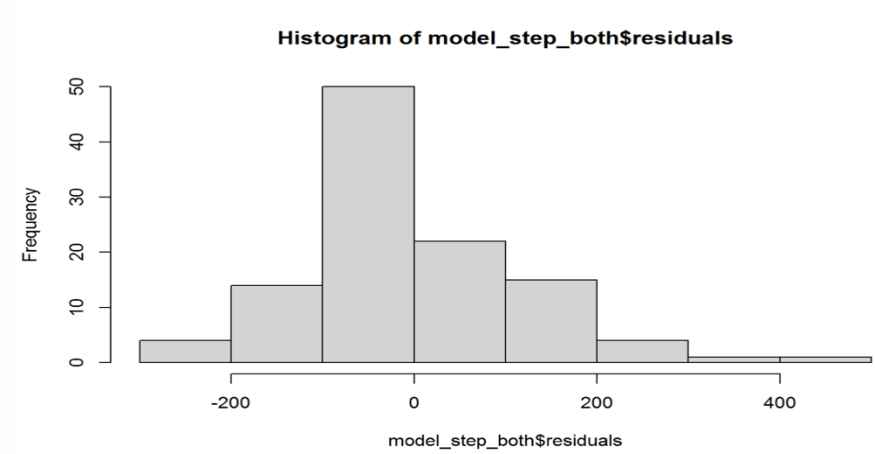
We will conduct Shapiro-Wilk Normality Test by using shapiro.test() function.
Hypothesis:

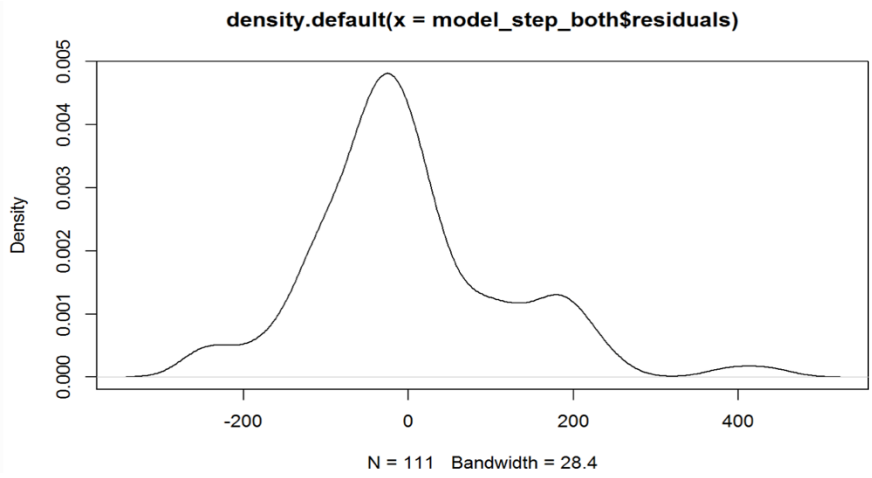
H_0: The residuals follow normal distribution
H_1: The residuals does not follow normal distribution

```
> #normality test
> shapiro.test(model_step_both$residual

        Shapiro-Wilk normality test

data:  model_step_both$residuals
W = 0.95116, p-value = 0.0004733
```

From the result above, we can see that our p-value < 0.05, thus we reject the null hypothesis and our residuals are not following normal distribution.

We can also see the residuals distribution by creating histogram and density plot:





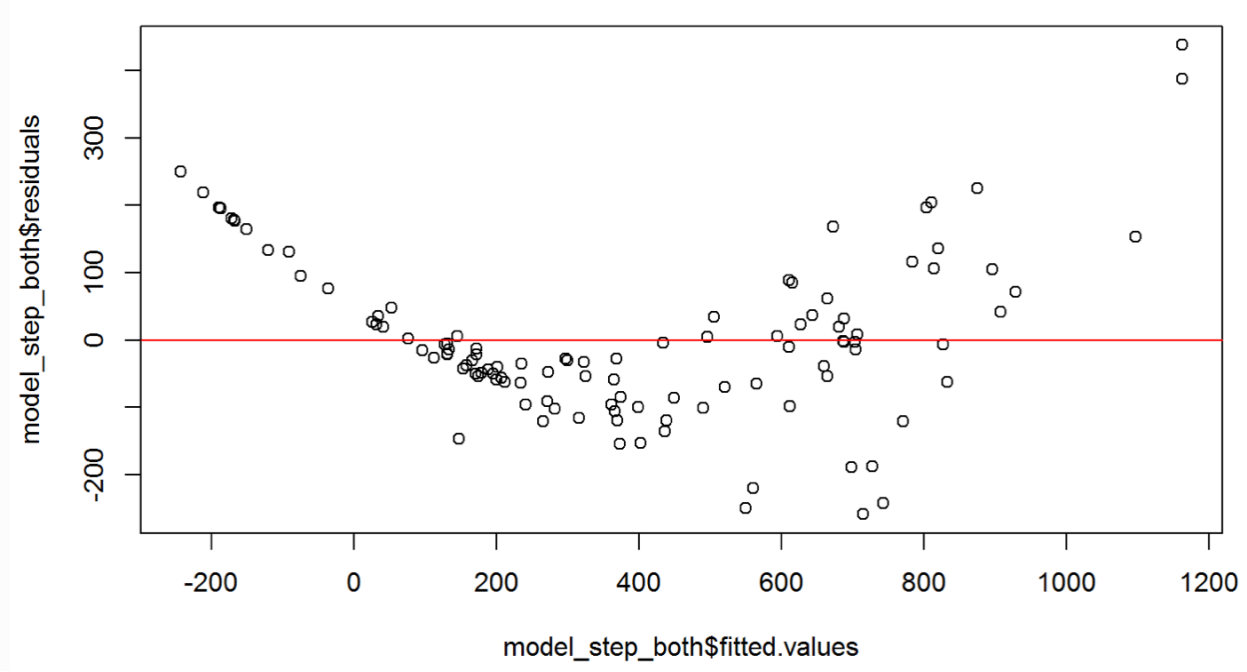
Homoscedasticity Test:

Hypothesis:

H_0 : Error variance distributed evently (Homoscedasticity)
H_1 : Error variance formed pattern (Heteroscedasticity)

```
> #Homoscedasticity Test
> bptest(model_step_both)

        studentized Breusch-Pagan test

data:  model_step_both
BP = 40.342, df = 3, p-value = 9.017e-09
```

From the above result, it can be seen that our value of p <0.05, so we reject the null hypothesis, means that our data have formed a pattern (heteroscedasticity), so we cannot follow the assumption of homoscedasticity.
Let's visualize the result using a scatter plot between the predicted values (fitted values) and the residuals.



Multicolinearity test:

The multicolinearity existence can be seen by the VIF (Variance Inflation Factor) value. VIF value shows how big the coefficient variance increase due to multicolinearity. VIF values greater than 10 indicate that there is multicolinearity in our data.
We conduct Multicolinearity test by using vif() function from car package.

```
> #Multicolinearity test
> vif(model_step_both)
  Length3    Height   Length1
207.26264   5.65118 171.31491
```

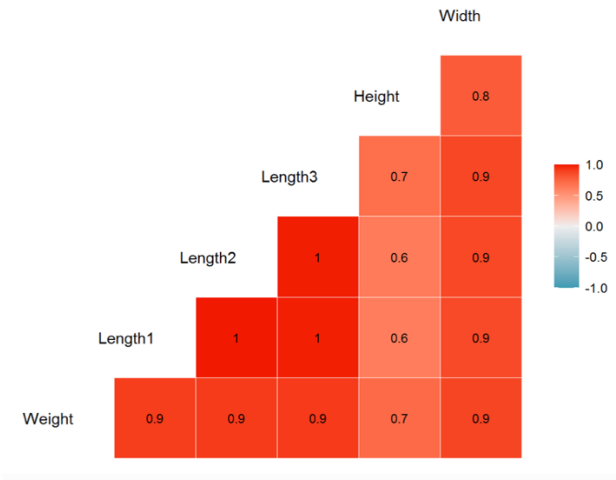
From the result above, we can see that our Length3 and Length1 variables have VIF value > 10, thus means we have multicolinearity in our data.

From the tests that we have been conducted, none of these assumptions met by our model. Next step, we will try to tune data to make sure we able to fulfill the assumptions needed to make sure that our model will not be misleading to predict our test dataset.

Data and Model tuning

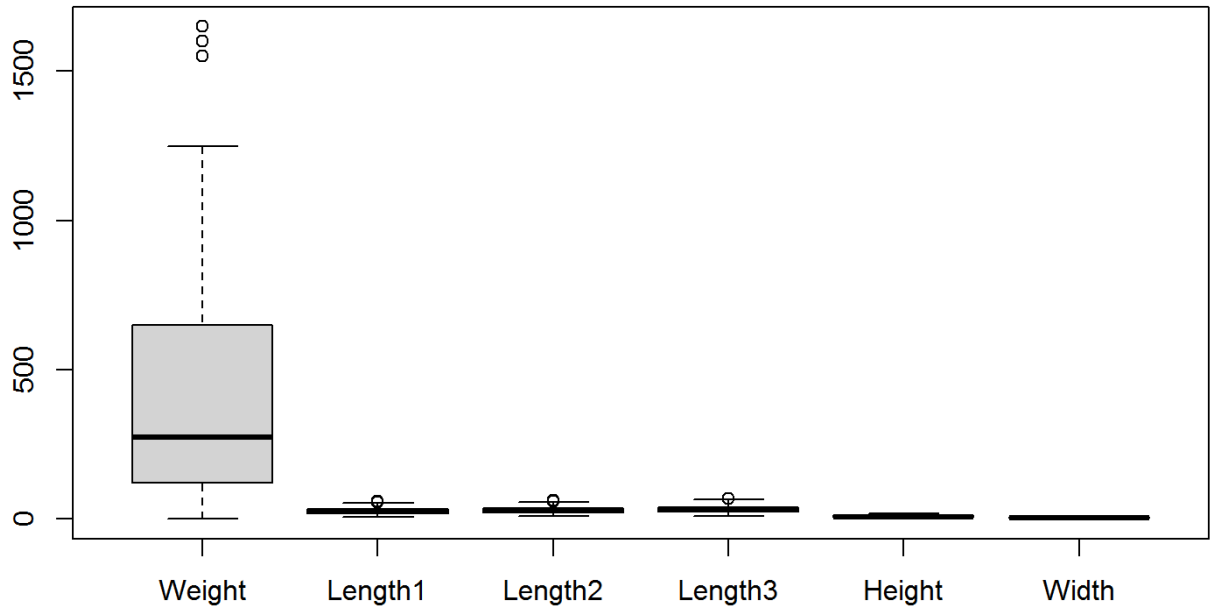
As can be seen from the assumption checking section, our model does not meet all the necessary assumptions and may mislead about the desired result. To fulfill the assumptions, we will first set up our model.

Let's take another look at the correlation between predictor variables.



From graphic above, we can see that the Length1 and Length2 variables are very dependent with Length3 variable. We will try to remove these variables from our dataset and try to recreate our model from new dataset.

Next, we will see if there is any outlier in our fish data.



From the boxplot chart, we can see that we have a few outliers in our data. Hence, we will remove the outlier from our fish data and store it in object called fish_new.

We will divide our new data into training and test dataset based on fish_new data. Next, we will create model based on fish_train_new data with Width as predictor variable and store it in model_init_new. Other models will be a model without

predictor variables which will be stored in model_null_new and a model with all predictor variables included, this model will be stored in model_all_new.
Again, these 3 models will be based on fish_train_new and will be different from our previous models.

```
# Comparison of Model Performance Indices

Name           | Model |      AIC | AIC weights |      BIC | BIC weights |    R2 | R2 (adj.) |    RMSE |   Sigma
-----------------------------------------------------------------------------------------------------------------
model_init_new |    lm | 1380.905 |     < 0.001 | 1388.979 |     < 0.001 | 0.830 |     0.828 | 132.683 | 133.918
model_null_new |    lm | 1572.095 |     < 0.001 | 1577.478 |     < 0.001 | 0.000 |     0.000 | 321.873 | 323.360
model_all_new  |    lm | 1346.753 |       1.000 | 1360.210 |       1.000 | 0.880 |     0.877 | 111.381 | 113.483
>
```

Here is a comparison with our previously created model. we can see that based on the AIC and adjusted R-square, the model_all_rev model was better than the other two models.
Next, we will create another model using the Step_Wise regression method and compare it with model_all_new.

```
# Comparison of Model Performance Indices

Name                | Model |      AIC | AIC weights |      BIC | BIC weights |    R2 | R2 (adj.) |    RMSE |   Sigma
------------------------------------------------------------------------------------------------------------------
model_step_back_new |    lm | 1346.753 |       0.250 | 1360.210 |       0.250 | 0.880 |     0.877 | 111.381 | 113.483
model_step_both_new |    lm | 1346.753 |       0.250 | 1360.210 |       0.250 | 0.880 |     0.877 | 111.381 | 113.483
model_step_forw_new |    lm | 1346.753 |       0.250 | 1360.210 |       0.250 | 0.880 |     0.877 | 111.381 | 113.483
model_all_new       |    lm | 1346.753 |       0.250 | 1360.210 |       0.250 | 0.880 |     0.877 | 111.381 | 113.483
>
```

All adjusted R-squares are about 0.88 or 88%. This means that both models that we created earlier can explain 88% of the variance of our target variable.Based on the comparison above, all these models have the same value for AIC and adjusted R-square. Thus, it will not be a problem for us to randomly choose between these models, and we will use model_step_both_new for further analysis.

Evaluation

```
fish_pred_new <- predict(object = model_step_both_new, newdata = fish_test_new, level = 0.95)

# RMSE of train_rev dataset
RMSE(y_pred = model_step_both_new$fitted.values, y_true = fish_train_new$Weight)
1] 111.381

# RMSE of test_rev dataset
RMSE(y_pred = fish_pred_new, y_true = fish_test_new$Weight)
1] 94.73077
```

Although our predictions in the train data are higher than the test data, the difference is not that big.
We can conclude that our model is good enough to predict the test dataset.

Assumpltion Checking: Based on Tuned data

Normality test:

H_0: The residuals follow normal distribution
H_1: The residuals does not follow normal distribution

```
        Shapiro-Wilk normality test

data:  model_step_both_new$residuals
W = 0.93915, p-value = 8.548e-05
```

From the result above, we can see that our p-value < 0.05, thus we reject the null hypothesis and our residuals are not following normal distribution.

Homoscedasticity Test:

Hypothesis:

H_0 : Error variance distributed evently (Homoscedasticity)
H_1 : Error variance formed pattern (Heteroscedasticity)

```
        studentized Breusch-Pagan test

data:  model_step_both_new
BP = 6.7147, df = 3, p-value = 0.08157
```

From the above result, it can be seen that our value of p > 0.05, therefore, we cannot reject the null hypothesis, means that our data have not formed a pattern (heteroscedasticity), and we fulfill the assumption of homoscedasticity.

Multicolinearity test:

```
> vif(model_step_both_new)
 Length3    Width   Height
6.084121 7.089598 3.124635
```

From the above result, we see that none of these variables has a value of VIF> 10, which means that we do not have multicollinearity in our data, and we satisfy the multicollinearity assumption.

Conclusion

Based on the analysis, we can conclude that predictor variables such as length3, width and height have a significant impact on our weight as our target variable. Our model is also capable of fulfilling two of the three classical assumptions. The adjusted R-squared of the model has a value of 0.876, which means that 87.6% of the variables can explain the fish weight variance. The accuracy of our model predicting fish weight can be measured using RMSE, the training data set has an RMSE of about 111.381, while our test data set has an RMSE of about 94.731. Although the RMSE of the training data has a higher value than our test data, the difference is not that big and we can assume that our model is good enough to predict the test data set.

References:

https://www.kaggle.com/datasets/aungpyaeap/fish-market

Eberly L.E. (2007) Multiple Linear Regression. In: Ambrosius W.T. (eds) Topics in Biostatistics. Methods in Molecular Biology™, vol 404. Humana Press. https://doi.org/10.1007/978-1-59745-530-5_9

Appendix:

R-Script (submitted along with this report)