mva_final_assessment

Thomas Frank

2024-10-02

Data Exploration

```
knitr::opts_chunk$set(echo = TRUE, eval = TRUE)
# Explore Data
library(GGally)
```

```
## Loading required package: ggplot2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(ggplot2)
library(readr)
abalone <- read_csv('/Users/macbook/Desktop/mva/Final Project/abalone.csv')
```

```
## Rows: 4177 Columns: 9
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Sex
## dbl (8): Length, Diameter, Height, Whole weight, Shucked weight, Viscera wei...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(abalone)
```

```
## # A tibble: 6 x 9
##   Sex   Length Diameter Height `Whole weight` `Shucked weight` `Viscera weight`
##   <chr>  <dbl>    <dbl>  <dbl>          <dbl>            <dbl>             <dbl>
## 1 M         91       73     19          103.              44.9              20.2
## 2 M         70       53     18           45.1             19.9               9.7
## 3 F        106       84     27          135.              51.3              28.3
## 4 M         88       73     25          103.              43.1              22.8
## 5 I         66       51     16           41               17.9               7.9
## 6 I         85       60     19           70.3             28.2              15.5
## # i 2 more variables: `Shell weight` <dbl>, Rings <dbl>
```

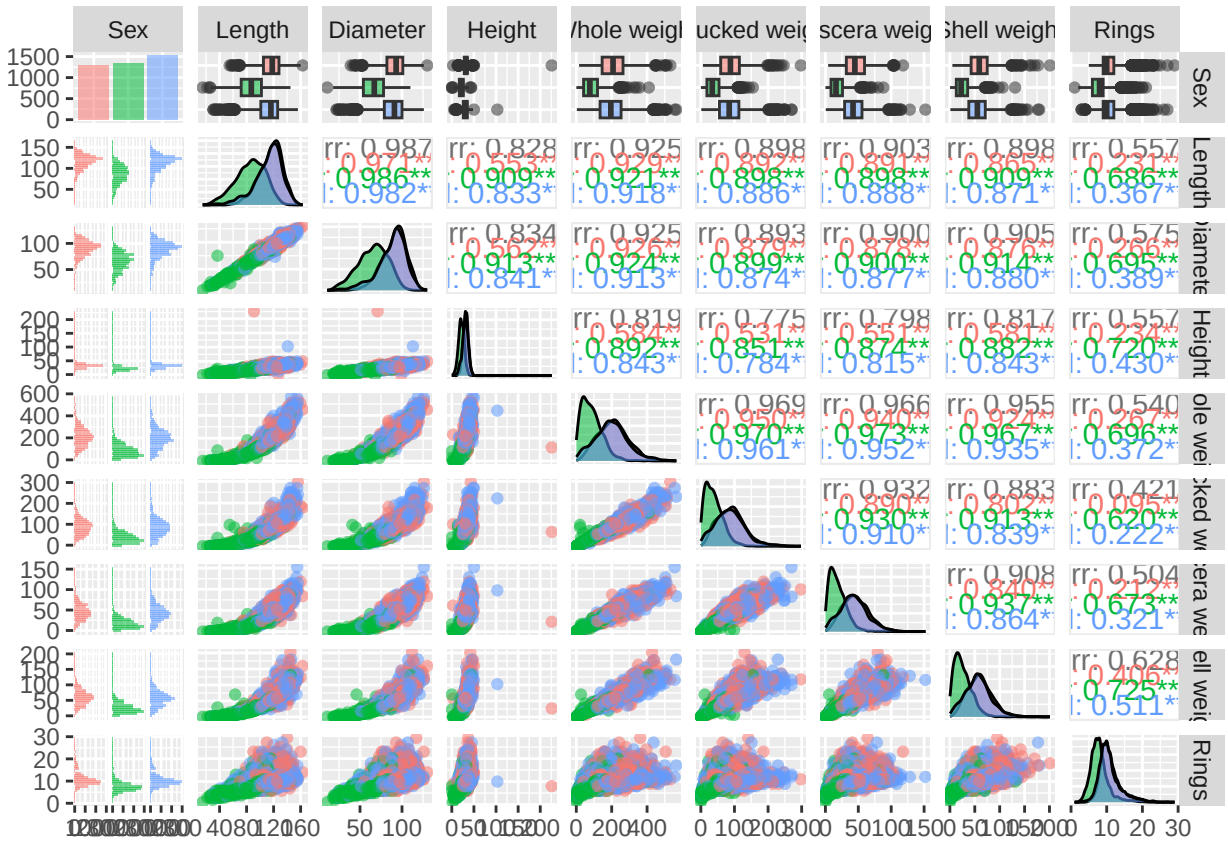
```
summary(abalone)
```

```
##      Sex                Length         Diameter          Height
##  Length:4177        Min.   : 15.0   Min.   : 11.00   Min.   :  0.0
##  Class :character   1st Qu.: 90.0   1st Qu.: 70.00   1st Qu.: 23.0
##  Mode  :character   Median :109.0   Median : 85.00   Median : 28.0
```

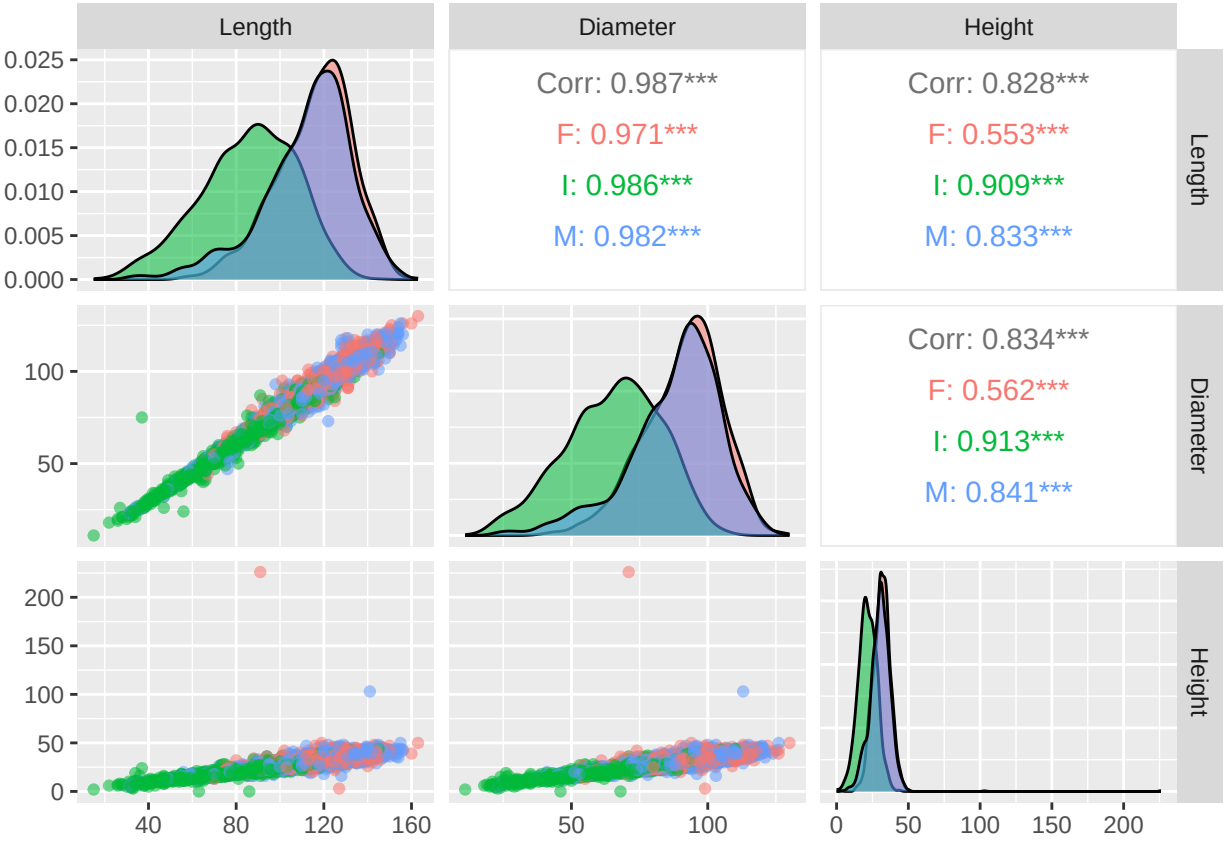
```
##                    Mean   :104.8   Mean   : 81.58   Mean   : 27.9
##                    3rd Qu.:123.0   3rd Qu.: 96.00   3rd Qu.: 33.0
##                    Max.   :163.0   Max.   :130.00   Max.   :226.0
##   Whole weight   Shucked weight   Viscera weight    Shell weight
##  Min.   :  0.4   Min.   :  0.20   Min.   :  0.10   Min.   :  0.30
##  1st Qu.: 88.3   1st Qu.: 37.20   1st Qu.: 18.70   1st Qu.: 26.00
##  Median :159.9   Median : 67.20   Median : 34.20   Median : 46.80
##  Mean   :165.7   Mean   : 71.87   Mean   : 36.12   Mean   : 47.77
##  3rd Qu.:230.6   3rd Qu.:100.40   3rd Qu.: 50.60   3rd Qu.: 65.80
##  Max.   :565.1   Max.   :297.60   Max.   :152.00   Max.   :201.00
##      Rings
##  Min.   : 1.000
##  1st Qu.: 8.000
##  Median : 9.000
##  Mean   : 9.934
##  3rd Qu.:11.000
##  Max.   :29.000
```

```
# Data Visualization
ggpairs(abalone, mapping = aes(colour = Sex, alpha = 0.3))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggpairs(abalone, mapping = aes(colour = Sex, alpha = 0.3), columns = c("Length", "Diameter", "Height"))
```



Infant abalones are likely to be more easily distinguishable from adult abalones based on their exterior measurements - they cluster at the lower end of the data set while Male and Female abalones are more challenging to separate as their measurements overlap significantly in the upper ranges.

1_1. Predicting the Sex of the Abalone

```
########################################
#Method 1 = Using Discriminant Analysis
########################################
#LDA and QDA classify abalone based on exterior measurements by finding
#boundaries between different sexes. LDA assumes equal covariance across classes
#QDA allows for varying covariances, offering flexibility in handling the
#differences between male, female, and infant abalones in the abalone dataset.
library(MASS)
abalone$Sex <- as.factor(abalone$Sex)
fit_lda <- lda(Sex ~ Length + Diameter + Height, data=abalone)
fit_qda <- qda(Sex ~ Length + Diameter + Height, data=abalone)
summary(fit_lda)
```

```
##         Length Class  Mode
## prior   3      -none- numeric
## counts  3      -none- numeric
## means   9      -none- numeric
## scaling 6      -none- numeric
## lev     3      -none- character
## svd     2      -none- numeric
## N       1      -none- numeric
```

```
## call    3      -none- call
## terms   3      terms  call
## xlevels 0      -none- list
```

```
summary(fit_qda)
```

```
##         Length Class  Mode
## prior    3     -none- numeric
## counts   3     -none- numeric
## means    9     -none- numeric
## scaling 27     -none- numeric
## ldet     3     -none- numeric
## lev      3     -none- character
## N        1     -none- numeric
## call     3     -none- call
## terms    3     terms  call
## xlevels  0     -none- list
```

```
predictions_lda <- predict(fit_lda, abalone)$class
predictions_qda <- predict(fit_qda, abalone)$class

# Confusion matrices
confusion_matrix_lda <- table(Predicted = predictions_lda, Actual = abalone$Sex)
confusion_matrix_qda <- table(Predicted = predictions_qda, Actual = abalone$Sex)

#Accuracy
accuracy_lda <- sum(diag(confusion_matrix_lda)) / sum(confusion_matrix_lda)
accuracy_qda <- sum(diag(confusion_matrix_qda)) / sum(confusion_matrix_qda)

#Method 2 = Using SVM
######################
#Has a linear kernel and is useful for the abalone dataset as it aims to identify
#optimal boundaries between abalone sexes, which is valuable when the
#relationship between exterior measurements and sex classification is complex.
library(e1071)
fit_svm <- svm(Sex ~ Length + Diameter + Height, data=abalone, kernel="linear")
summary(fit_svm)
```

```
##
## Call:
## svm(formula = Sex ~ Length + Diameter + Height, data = abalone, kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  3651
##
##  ( 1528 1307 816 )
##
##
## Number of Classes:  3
##
```

```
## Levels:
##  F I M
```

```
predictions <- predict(fit_svm, abalone)

# Confusion Matrix
table(Predicted = predictions, Actual = abalone$Sex)
```

```
##          Actual
## Predicted    F    I    M
##         F    0    0    0
##         I  202  974  310
##         M 1105  368 1218
```

```
confusion_matrix_svm <- table(Predicted = predictions, Actual = abalone$Sex)
accuracy_svm <- sum(diag(confusion_matrix_svm)) / sum(confusion_matrix_svm)
# Summary Table of Accuracy
summary_table <- data.frame(
  Method = c("LDA", "QDA", "SVM"),
  Accuracy = c(accuracy_lda, accuracy_qda, accuracy_svm)
)

#print results to compare models
print(confusion_matrix_lda)
```

```
##          Actual
## Predicted   F   I   M
##         F 297   9 290
##         I 182 931 291
##         M 828 402 947
```

```
print(confusion_matrix_qda)
```

```
##          Actual
## Predicted    F    I    M
##         F  190   52  183
##         I  204  978  324
##         M  913  312 1021
```

```
print(confusion_matrix_svm)
```

```
##          Actual
## Predicted    F    I    M
##         F    0    0    0
##         I  202  974  310
##         M 1105  368 1218
```

```
print(summary_table)
```

```
##   Method  Accuracy
## 1    LDA 0.5207086
## 2    QDA 0.5240603
## 3    SVM 0.5247785
```

SUMMARY:

Moderate prediction accuracies of around 52% for LDA and QDA. Similar for SVM, however it misclassified all females.Suggests methods aren't great at classifying all sexes together.

1_2. Binary Classification for Infants (to avoid harvesting them)

```
# Binary Classification: LDA
abalone$Infant_Other <- ifelse(abalone$Sex == "I", "Infant", "Other")
fit_lda_infant <- lda(Infant_Other ~ Length + Diameter + Height, data = abalone)
predictions_lda_infant <- predict(fit_lda_infant)$class
confusion_matrix_lda_infant <- table(Predicted = predictions_lda_infant, Actual = abalone$Infant_Other)
accuracy_lda_infant <- sum(diag(confusion_matrix_lda_infant)) / sum(confusion_matrix_lda_infant)
#1_3_QDA
fit_qda_infant <- qda(Infant_Other ~ Length + Diameter + Height, data = abalone)
predictions_qda_infant <- predict(fit_qda_infant)$class
confusion_matrix_qda_infant <- table(Predicted = predictions_qda_infant, Actual = abalone$Infant_Other)
accuracy_qda_infant <- sum(diag(confusion_matrix_qda_infant)) / sum(confusion_matrix_qda_infant)
# Binary Classification: SVM
abalone$Infant_Other <- as.factor(abalone$Infant_Other)
library(e1071)
fit_svm_infant <- svm(Infant_Other ~ Length + Diameter + Height, data = abalone, kernel = "linear")
predictions_svm_infant <- predict(fit_svm_infant, abalone)
confusion_matrix_svm_infant <- table(Predicted = predictions_svm_infant, Actual = abalone$Infant_Other)
accuracy_svm_infant <- sum(diag(confusion_matrix_svm_infant)) / sum(confusion_matrix_svm_infant)
# Summary Table of Accuracy
summary_table_infant <- data.frame(
  Method = c("LDA", "QDA", "SVM"),
  Accuracy = c(accuracy_lda_infant, accuracy_qda_infant, accuracy_svm_infant)
)
#Output results
print(confusion_matrix_lda_infant)
```

```
##          Actual
## Predicted Infant Other
##    Infant    757   276
##    Other     585  2559
```

```
print(confusion_matrix_qda_infant)
```

```
##          Actual
## Predicted Infant Other
##    Infant    871   397
##    Other     471  2438
```

```
print(confusion_matrix_svm_infant)
```

```
##          Actual
## Predicted Infant Other
##    Infant    783   296
##    Other     559  2539
```

```
print(summary_table_infant)
```

```
##   Method  Accuracy
## 1    LDA 0.7938712
```

```
## 2    QDA 0.7921954
## 3    SVM 0.7953076
```

SUMMARY:

Great prediction accuracies of around 79% for each method. results suggest that discriminating infants from other abalone based on size measurements is feasible with fairly high accuracy.

1_3. BINARY CLASSIFICATION For FEMALES (when profitability is prioritised)

```
#1_3_LDA
abalone$Female_Other <- ifelse(abalone$Sex == "F", "Female", "Other")
fit_lda_female <- lda(Female_Other ~ Length + Diameter + Height, data = abalone)
predictions_lda_female <- predict(fit_lda_female)$class
confusion_matrix_lda_female <- table(Predicted = predictions_lda_female, Actual = abalone$Female_Other)
accuracy_lda_female <- sum(diag(confusion_matrix_lda_female)) / sum(confusion_matrix_lda_female)
#1_3_QDA
fit_qda_female <- qda(Female_Other ~ Length + Diameter + Height, data = abalone)
predictions_qda_female <- predict(fit_qda_female)$class
confusion_matrix_qda_female <- table(Predicted = predictions_qda_female, Actual = abalone$Female_Other)
accuracy_qda_female <- sum(diag(confusion_matrix_qda_female)) / sum(confusion_matrix_qda_female)
#1_3_SVM
abalone$Female_Other <- as.factor(abalone$Female_Other)
fit_svm_female <- svm(Female_Other ~ Length + Diameter + Height, data = abalone, kernel = "linear")
predictions_svm_female <- predict(fit_svm_female, abalone)
confusion_matrix_svm_female <- table(Predicted = predictions_svm_female, Actual = abalone$Female_Other)
accuracy_svm_female <- sum(diag(confusion_matrix_svm_female)) / sum(confusion_matrix_svm_female)
# Summary Table of Accuracy
summary_table_female <- data.frame(
  Method = c("LDA", "QDA", "SVM"),
  Accuracy = c(accuracy_lda_female, accuracy_qda_female, accuracy_svm_female)
)
#Output results
print(confusion_matrix_lda_female)
```

```
##          Actual
## Predicted Female Other
##    Female    236   223
##    Other    1071  2647
```

```
print(confusion_matrix_qda_female)
```

```
##          Actual
## Predicted Female Other
##    Female    192   218
##    Other    1115  2652
```

```
print(confusion_matrix_svm_female)
```

```
##          Actual
## Predicted Female Other
##    Female      0     0
##    Other    1307  2870
```

```r
print(summary_table_female)
```

```
##   Method  Accuracy
## 1    LDA 0.6902083
## 2    QDA 0.6808714
## 3    SVM 0.6870960
```

SUMMARY:

Good prediction accuracies of around 68-69% for LDA and QDA. Similar for SVM, however it misclassified all females.

1_4. BINARY CLASSIFICATION For MALE (when sustainability is prioritised)

```r
#1_4_LDA
abalone$Male_Other <- ifelse(abalone$Sex == "M", "Male", "Other")
fit_lda_male <- lda(Male_Other ~ Length + Diameter + Height, data = abalone)
predictions_lda_male <- predict(fit_lda_male)$class
confusion_matrix_lda_male <- table(Predicted = predictions_lda_male, Actual = abalone$Male_Other)
accuracy_lda_male <- sum(diag(confusion_matrix_lda_male)) / sum(confusion_matrix_lda_male)
#1_4_QDA
fit_qda_male <- qda(Male_Other ~ Length + Diameter + Height, data = abalone)
predictions_qda_male <- predict(fit_qda_male)$class
confusion_matrix_qda_male <- table(Predicted = predictions_qda_male, Actual = abalone$Male_Other)
accuracy_qda_male <- sum(diag(confusion_matrix_qda_male)) / sum(confusion_matrix_qda_male)
#1_4_SVM
abalone$Male_Other <- as.factor(abalone$Male_Other)
fit_svm_male <- svm(Male_Other ~ Length + Diameter + Height, data = abalone, kernel = "linear")
predictions_svm_male <- predict(fit_svm_male, abalone)
confusion_matrix_svm_male <- table(Predicted = predictions_svm_male, Actual = abalone$Male_Other)
accuracy_svm_male <- sum(diag(confusion_matrix_svm_male)) / sum(confusion_matrix_svm_male)
# Summary Table of Accuracy
summary_table_male <- data.frame(
  Method = c("LDA", "QDA", "SVM"),
  Accuracy = c(accuracy_lda_male, accuracy_qda_male, accuracy_svm_male)
)
print(confusion_matrix_lda_male)
```

```
##          Actual
## Predicted Male Other
##     Male   225   251
##     Other 1303  2398
```

```r
print(confusion_matrix_qda_male)
```

```
##          Actual
## Predicted Male Other
##     Male   665   697
##     Other  863  1952
```

```r
print(confusion_matrix_svm_male)
```

```
##          Actual
## Predicted Male Other
##     Male     0     0
##     Other 1528  2649
```

```r
print(summary_table_male)
```

```
##   Method  Accuracy
## 1    LDA 0.6279627
## 2    QDA 0.6265262
## 3    SVM 0.6341872
```

SUMMARY:

Moderate prediction accuracies of around 63% for LDA and QDA. Similar for SVM, however it misclassified all males.

QUESTION 1 CONCLUSION:

LDA and QDA are suitable for predicting the sex of abalone, with a particular strength in identifying infants, which is crucial for sustainable harvesting. While the models achieve moderate success in classifying males and females, further refinement or different techniques are recommended for more accurate predictions.

Question 2

2__1. Produce the best model for predicting Viscera and Shucked Weight:

Test four different multivariate models to predict shucked and visceral weights based on the abalone dataset. We will test 4 that are likely to work well: Multivariate Linear Model (MLM), Log-Transformed Model, Polynomial Model, Interaction Model (Selected Model)

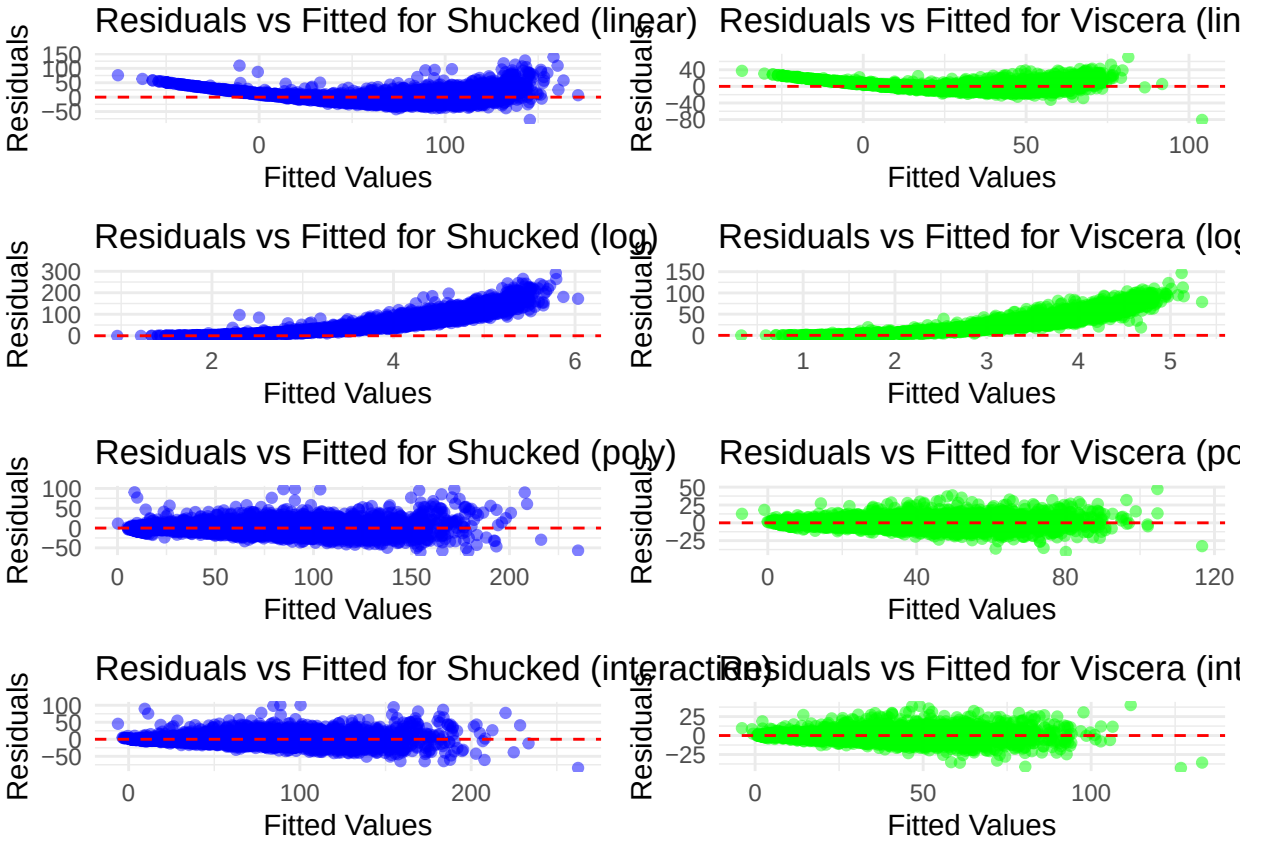
```r
library(gridExtra)
library(Metrics)

#2_1. Produce the best model for predicting Viscera and Shucked Weight
models <- list()

# Multivariate Linear Model
fit_mlm <- lm(cbind(`Shucked weight`, `Viscera weight`) ~ Length + Diameter + Height, data = abalone)
models$linear <- fit_mlm

# Multivariate LOG Model
fit_mlm_log <- lm(cbind(log(`Shucked weight`), log(`Viscera weight`)) ~ Length + Diameter + Height, data
models$log <- fit_mlm_log

# Multivariate Polynomial Model
fit_mlm_poly <- lm(cbind(`Shucked weight`, `Viscera weight`) ~ poly(Length, 2) + poly(Diameter, 2) + pol
models$poly <- fit_mlm_poly

# Interaction Model
fit_mlm_interaction <- lm(cbind(`Shucked weight`, `Viscera weight`) ~ Length * Diameter * Height, data =
models$interaction <- fit_mlm_interaction
```

```r
# Function to get residuals and generate ggplot
plot_residuals_vs_fitted <- function(model, model_name) {
  predicted_values <- as.data.frame(predict(model))

  abalone$Residuals_Shucked <- abalone$`Shucked weight` - predicted_values[, 1]
  abalone$Residuals_Viscera <- abalone$`Viscera weight` - predicted_values[, 2]

  p1 <- ggplot(abalone, aes(x = predicted_values[, 1], y = Residuals_Shucked)) +
    geom_point(color = "blue", alpha = 0.5) +
    geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
    labs(title = paste("Residuals vs Fitted for Shucked (", model_name, ")", sep=""), x = "Fitted Values
    theme_minimal()

  p2 <- ggplot(abalone, aes(x = predicted_values[, 2], y = Residuals_Viscera)) +
    geom_point(color = "green", alpha = 0.5) +
    geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
    labs(title = paste("Residuals vs Fitted for Viscera (", model_name, ")", sep=""), x = "Fitted Values
    theme_minimal()

  return(list(p1, p2))
}

# Error Metrics for RMSE and R2
error_metrics <- function(model, model_name) {
  predicted_values <- as.data.frame(predict(model))

  rmse_shucked <- rmse(abalone$`Shucked weight`, predicted_values[, 1])
  rmse_viscera <- rmse(abalone$`Viscera weight`, predicted_values[, 2])

  r2_shucked <- summary(lm(abalone$`Shucked weight` ~ predicted_values[, 1]))$r.squared
  r2_viscera <- summary(lm(abalone$`Viscera weight` ~ predicted_values[, 2]))$r.squared

  cat("\nModel: ", model_name)
  cat("\nRMSE Shucked:", rmse_shucked, "\nRMSE Viscera:", rmse_viscera)
  cat("\nR-squared Shucked:", r2_shucked, "\nR-squared Viscera:", r2_viscera)
  cat("\n------------------------------")
}

# Output plots and error metrics
plot_list <- list()
for (name in names(models)) {
  plot_list <- c(plot_list, plot_residuals_vs_fitted(models[[name]], name))
  error_metrics(models[[name]], name)
}
```

```
##
## Model:  linear
## RMSE Shucked: 19.31654
## RMSE Viscera: 9.168356
## R-squared Shucked: 0.8106168
## R-squared Viscera: 0.8250585
## ------------------------------
## Model:  log
```

```
## RMSE Shucked: 80.69033
## RMSE Viscera: 39.04218
## R-squared Shucked: 0.8098055
## R-squared Viscera: 0.8237192
## ----------------------------
## Model:  poly
## RMSE Shucked: 14.71611
## RMSE Viscera: 6.899468
## R-squared Shucked: 0.8900819
## R-squared Viscera: 0.9009302
## ----------------------------
## Model:  interaction
## RMSE Shucked: 14.51208
## RMSE Viscera: 6.909523
## R-squared Shucked: 0.8931088
## R-squared Viscera: 0.9006413
## ----------------------------
```

```
do.call("grid.arrange", c(plot_list, ncol = 2))
```



MODEL SELECTION: Interaction model had the highest R^2, lowest RMSE and residuals and the residuals were spread at random with no patterns (unlike mlm and log). Therefore, we: #Select interaction model for best fit.#

2_2: Pre-compute Coefficients from the selected Model (interaction model):

This ensures fast, efficient predictions using precomputed summaries, meeting the algorithm's computational constraints without the need to retrain the model.

```
# summaries/coefficients from the fitted model
coefficients <- coef(fit_mlm_interaction)
print(coefficients)
```

```
##                         Shucked weight Viscera weight
## (Intercept)              -3.0055524818   5.0546772132
## Length                   -1.1430641976  -0.7941702154
## Diameter                  1.2333416122   0.8701605566
## Height                    1.7551848970  -0.0547869048
## Length:Diameter           0.0086823246   0.0033460633
## Length:Height             0.0306824562   0.0294718466
## Diameter:Height          -0.0951758451  -0.0446976519
## Length:Diameter:Height    0.0003664337   0.0001134365
```

```
# calc sigma
residuals <- residuals(fit_mlm_interaction)
df_residual <- df.residual(fit_mlm_interaction)
sigma <- sqrt(sum(residuals^2) / df_residual)
print(sigma)
```

```
## [1] 16.08843
```

2_3: Use Precomputed Coefficients for Prediction

```
# Function to predict value
predict_abalone_value_with_precomputed <- function(length, diameter, height, vshucked, vviscera, coeffi
  X_shucked <- coefficients[1, 1] + coefficients[2, 1] * length + coefficients[3, 1] * diameter + coeff
  X_viscera <- coefficients[1, 2] + coefficients[2, 2] * length + coefficients[3, 2] * diameter + coeff
  # the value formula for calculating
  estimated_value <- (vshucked * X_shucked) + (vviscera * X_viscera)

  t_value <- qt(1 - (1 - level) / 2, df = nrow(abalone) - 4)

  lower_bound_value <- (vshucked * (X_shucked - t_value * sigma)) + (vviscera * (X_viscera - t_value *
  upper_bound_value <- (vshucked * (X_shucked + t_value * sigma)) + (vviscera * (X_viscera + t_value *

  if (!is.numeric(lower_bound_value) || !is.numeric(upper_bound_value)) {
    lower_bound_value <- NA
    upper_bound_value <- NA
  }

  return(list(
    shucked_weight = X_shucked,
    viscera_weight = X_viscera,
    value = estimated_value,
    lower_bound = lower_bound_value,
    upper_bound = upper_bound_value
  ))
```

```
}
```

2__4. Example usage with an example single instance (the numbers can be changed as necessary):

NOTE: The below assumes a predicted value of v(shucked) = 10 and v(viscera) = 5, however there are no prices given so these are completely arbitrary numbers and can be adjusted according to the actual price.

```
# Example usage of the function
length <- 121
diameter <- 100
height <- 32
vshucked <- 10  #Example dollar value per gram of shucked weight - arbitrary number
vviscera <- 5   #Example dollar value per gram of viscera weight - arbitrary number

result <- predict_abalone_value_with_precomputed(length, diameter, height, vshucked, vviscera, coefficie
print(result)
```

```
## $shucked_weight
## [1] 38.18376
##
## $viscera_weight
## [1] -5.777044
##
## $value
## [1] 352.9524
##
## $lower_bound
## [1] -44.08247
##
## $upper_bound
## [1] 749.9872
```

2__5. Example usage on the entire Dataset using the Function:

Or it can be applied to an entire dataset as shown below to the entire Abalone dataset as an example.

```
predicted_value <- numeric(nrow(abalone))
lower_bound_value <- numeric(nrow(abalone))
upper_bound_value <- numeric(nrow(abalone))

# Loop through each row of the abalone dataset
for (i in 1:nrow(abalone)) {
  length <- abalone$Length[i]
  diameter <- abalone$Diameter[i]
  height <- abalone$Height[i]
  vshucked <- 10  #Example dollar value per gram of shucked weight - arbitrary number
  vviscera <- 5   #Example dollar value per gram of viscera weight - arbitrary number

  result <- predict_abalone_value_with_precomputed(length, diameter, height, vshucked, vviscera, coeffi
  #predict
  predicted_value[i] <- result$value
  lower_bound_value[i] <- result$lower_bound
```

```
  upper_bound_value[i] <- result$upper_bound
}

#results df
results_df <- data.frame(
  Length = abalone$Length,
  Diameter = abalone$Diameter,
  Height = abalone$Height,
  Predicted_Value = predicted_value,
  Lower_Bound_Value = lower_bound_value,
  Upper_Bound_Value = upper_bound_value
)


#Example usage on the entire dataset - Visualize the Results for the entire dataset
# Plot 1: Predicted Value with Intervals vs Diameter
plot_diameter <- ggplot(results_df, aes(x = Diameter, y = Predicted_Value)) +
  geom_errorbar(aes(ymin = Lower_Bound_Value, ymax = Upper_Bound_Value), width = 0.2, color = "darkgray
  geom_point(color = "navy", alpha = 0.5) +
  geom_smooth(method = "loess", se = FALSE, color = "lightblue", linetype = "solid") +
  ylim(0, 1500) +
  labs(title = "Predicted Value vs Diameter", x = "Diameter", y = "Value") +
  theme_minimal()

# Plot 2: Predicted Value with Intervals vs Length
plot_length <- ggplot(results_df, aes(x = Length, y = Predicted_Value)) +
  geom_errorbar(aes(ymin = Lower_Bound_Value, ymax = Upper_Bound_Value), width = 0.2, color = "darkgray
  geom_point(color = "navy", alpha = 0.5) +
  geom_smooth(method = "loess", se = FALSE, color = "lightblue", linetype = "solid") +
  ylim(0, 1500) +
  labs(title = "Predicted Value vs Length", x = "Length", y = "Value") +
  theme_minimal()

# Plot 3: Predicted Value with Intervals vs Height
plot_height <- ggplot(results_df, aes(x = Height, y = Predicted_Value)) +
  geom_errorbar(aes(ymin = Lower_Bound_Value, ymax = Upper_Bound_Value), width = 0.2, color = "darkgray
  geom_point(color = "navy", alpha = 0.5) +
  geom_smooth(method = "loess", se = FALSE, color = "lightblue", linetype = "solid") +
  ylim(0, 1500) +
  xlim(0, 50) +
  labs(title = "Predicted Value vs Height", x = "Height", y = "Value") +
  theme_minimal()

# Output results on a grid
grid.arrange(plot_diameter, plot_length, plot_height, ncol = 3)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 61 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 61 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

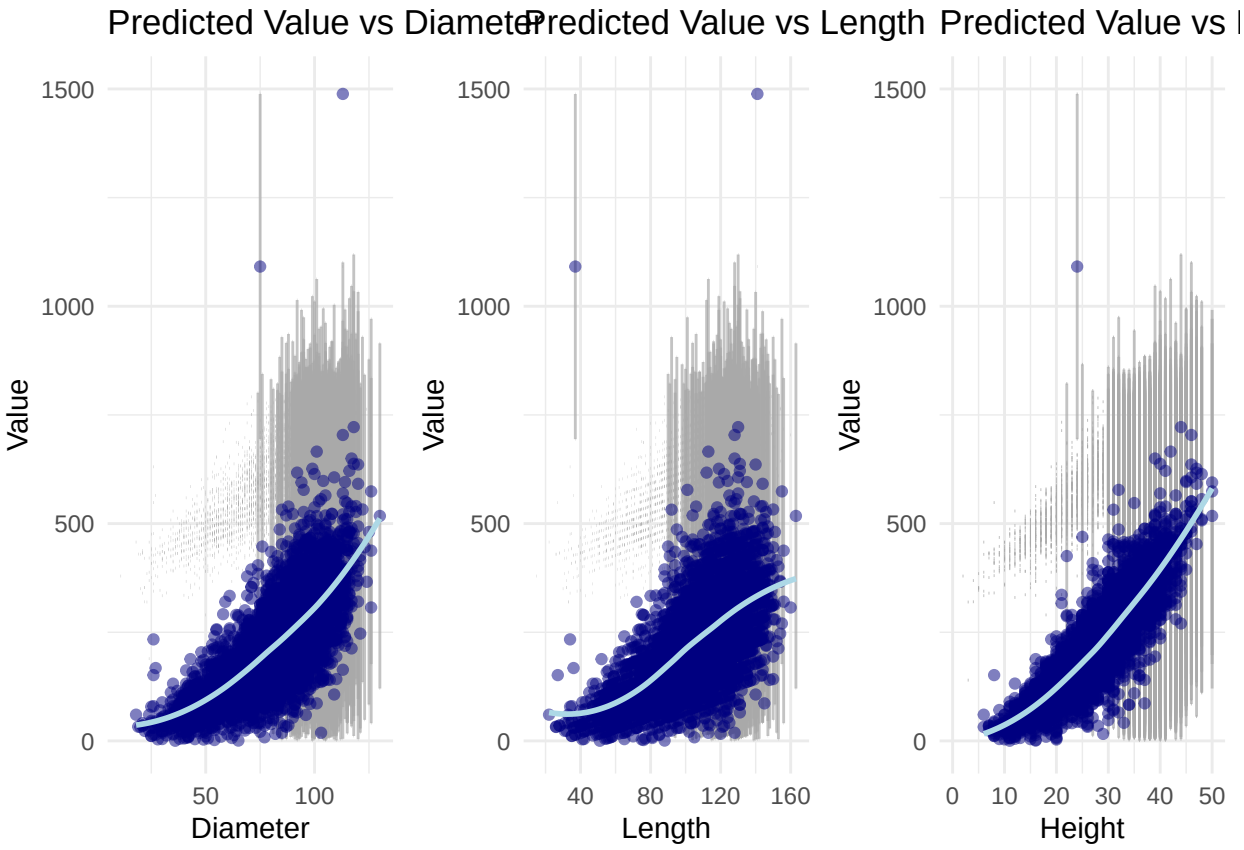
```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 61 rows containing non-finite outside the scale range
## (`stat_smooth()`).
## Removed 61 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 62 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 62 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



QUESTION 2 CONCLUSION: The interaction model was the best choice for predicting abalone weights. The use of pre-computed coefficients enables fast predictions, while prediction intervals offer insights into potential variability. This approach meets the computational requirements and provides reliable predictions for abalone profitability in dynamic market environments.