Palmer Penguins - Data Analysis & Modelling

Arnab Panja

30/08/2020

This document analyses the Palmer Penguins data set and studies the different body features of the penguins of the Palmer Islands of Antarctica. The study also involves creating a two-level classification modelling of the gender/sex of the penguins based on the body features of the penguins.

1. Study of Body Features with Species and Gender/Sex

The body features of bill depth, bill length, flipper length and body mass and their relationships with the species of the Penguins and their Gender/Sex is visualized in the below two figures.

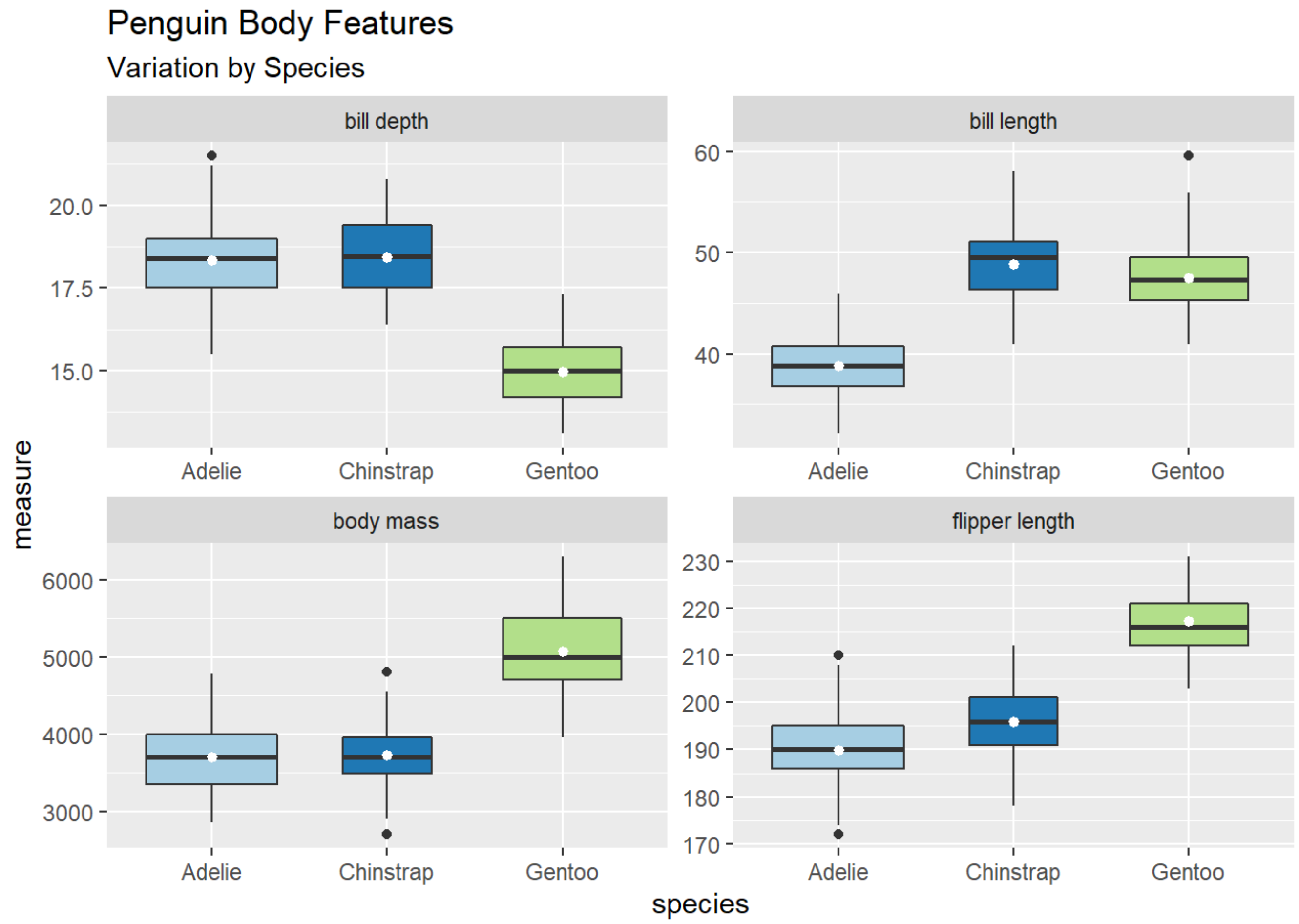
The graphs will indicate that there is a distinct relationship between a species and its body features such as bill depth, bill length, flipper length and body mass.

There also appears to be a relationship between the sex of the penguin and its body features though it may not be as prominent as the relationship with species. We will explore more on this relationship, build a classification model and study the accuracy of the model in the later sections of the document.

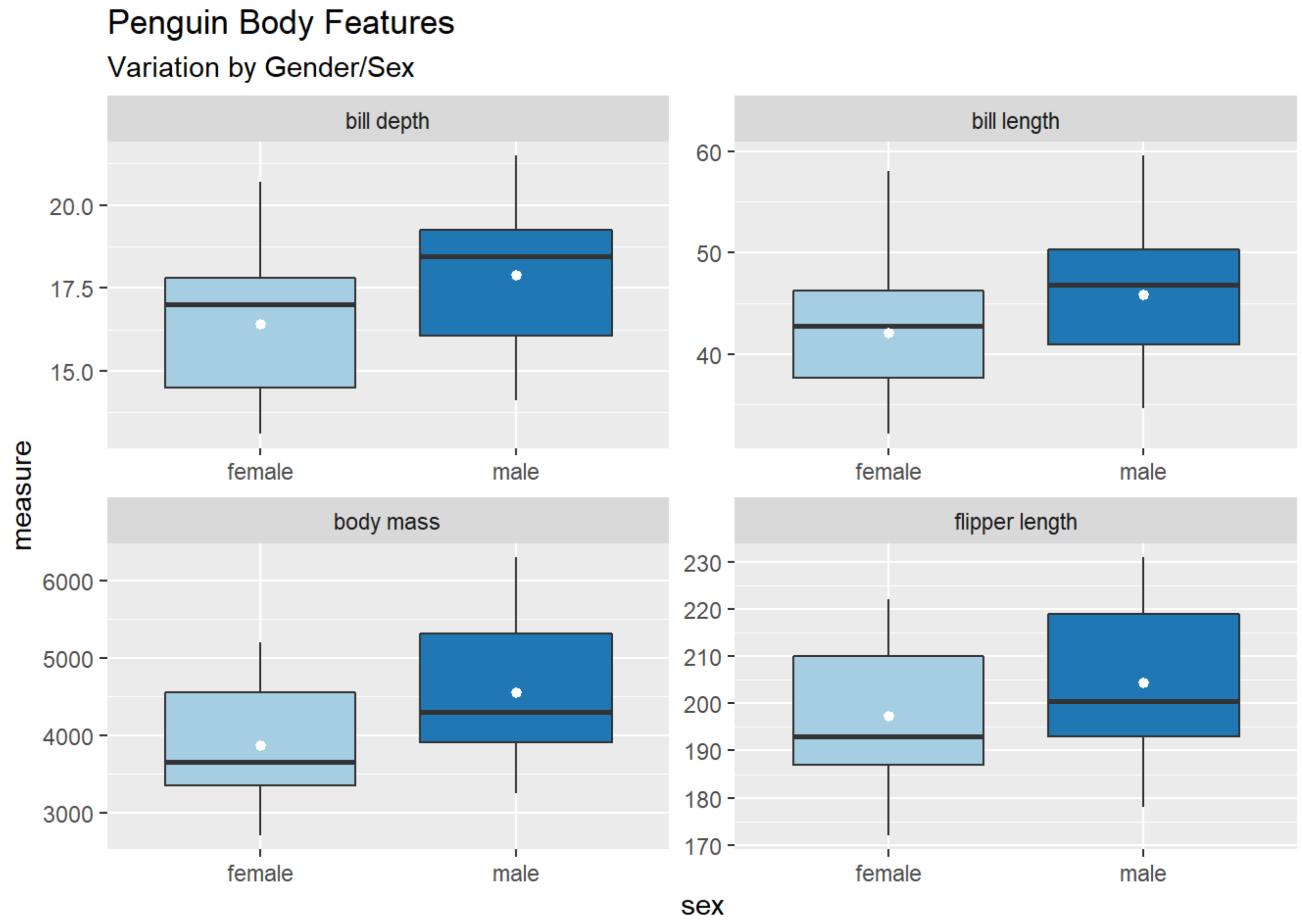
The data wrangling involved in the visualization is as below.

```
tbl_pivoted_penguins <- penguins %>% dplyr::select(species,
                                                   bill_length_mm,
                                                   bill_depth_mm,
                                                   flipper_length_mm,
                                                   body_mass_g,
                                                   sex) %>%
  pivot_longer(cols = c(bill_length_mm,
                        bill_depth_mm,
                        flipper_length_mm,
                        body_mass_g),
               names_to = "measure_type",
               values_to = "measure",
               values_drop_na = TRUE) %>%
  mutate(measure_type = case_when(str_detect(measure_type, "bill_length_mm") ~ "bill length",
                                  str_detect(measure_type, "bill_depth_mm") ~ "bill depth",
                                  str_detect(measure_type, "flipper_length_mm") ~ "flipper le
ngth",
                                  str_detect(measure_type, "body_mass_g") ~ "body mass",
                                  TRUE ~ measure_type))


p_variations_by_species <- ggplot(data = tbl_pivoted_penguins, mapping = aes(x = species, y =
measure)) +
  geom_boxplot(mapping = aes(fill = species),
               show.legend = FALSE,
               varwidth = TRUE,
               na.rm = TRUE) +
  stat_summary(fun = "mean", color = "white", na.rm = TRUE, size = 0.25) +
  facet_wrap(~measure_type, nrow = 2, scales = "free") +
  scale_fill_brewer(palette = "Paired") +
  theme_grey() +
  labs(title = "Penguin Body Features",
       subtitle = "Variation by Species")



p_variations_by_sex <- filter(tbl_pivoted_penguins, !is.na(sex)) %>% ggplot(mapping = aes(x =
sex, y = measure)) +
  geom_boxplot(mapping = aes(fill = sex),
               show.legend = FALSE,
               na.rm = TRUE,
               varwidth = TRUE) +
  stat_summary(fun = "mean", color = "white", na.rm = TRUE, size = 0.25) +
  scale_fill_brewer(palette = "Paired") +
  theme_grey() +
  facet_wrap(~measure_type, nrow = 2, scales = "free")  +
  labs(title = "Penguin Body Features",
       subtitle = "Variation by Gender/Sex")
```

The plots describing the variations of these body features with gender and sex are shown in the two plots below



The Gentoo penguins clearly features that measure different from the Adelie and the Chinstrap penguins.



The above graph indicates the male penguins have larger measurements on all four facets of the features.

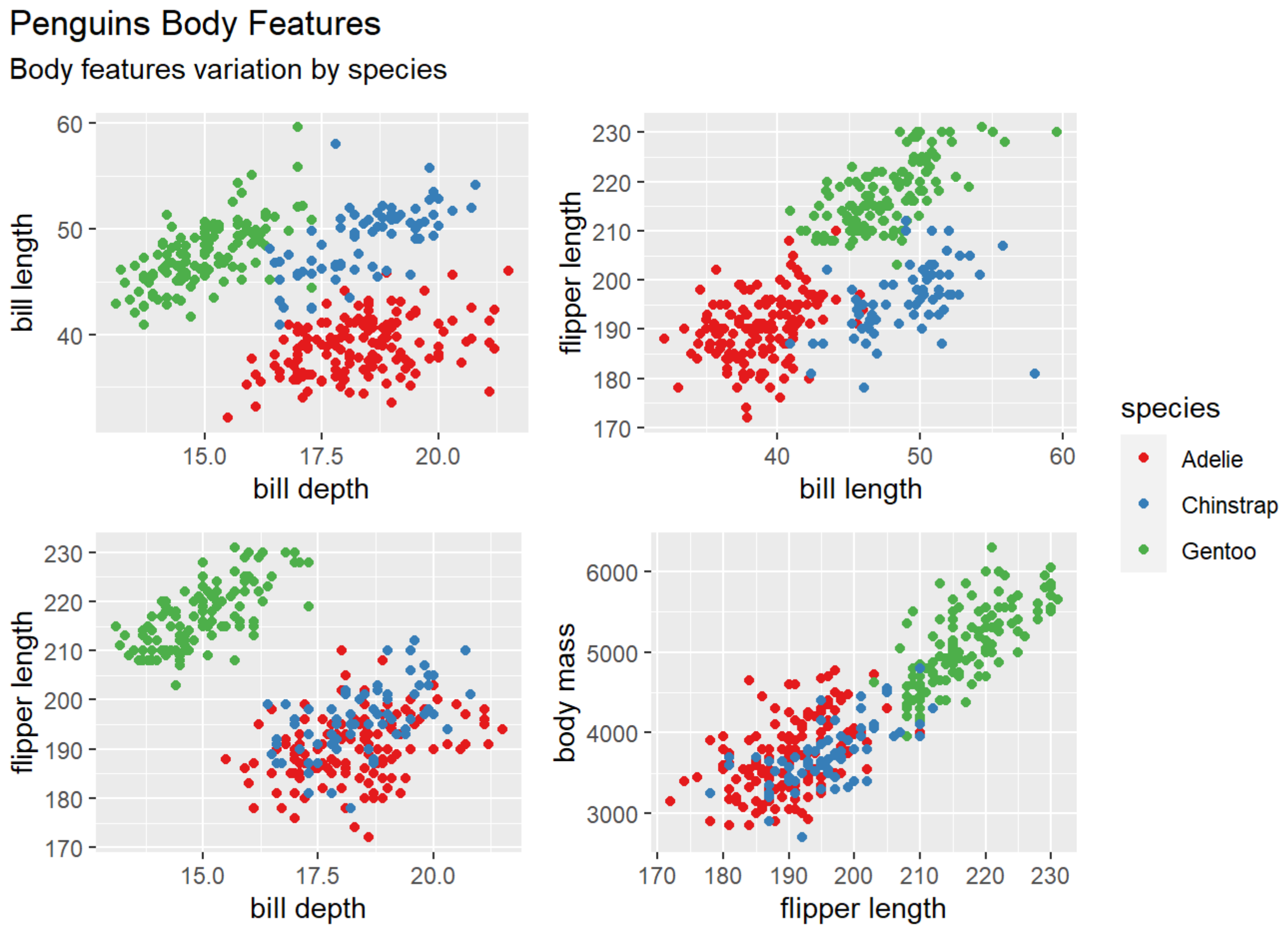
2. Study of Relationships between body features

The relationships of the body features also varies among the species of the penguins. The below set of code creates plots for bill depth, bill length, flipper length and body mass of the three penguin species and tries to identify that if given any of these features how far is it possible to identify which species the penguins belong to.

```
p1 <- filter(penguins, !is.na(sex)) %>%
  ggplot() +
  geom_point(mapping = aes(x = bill_depth_mm, y = bill_length_mm, color = species),
             na.rm = TRUE,
             show.legend = FALSE) +
  scale_color_brewer(palette = "Set1") +
  labs(x = "bill depth",
       y = "bill length")

p2 <- filter(penguins, !is.na(sex)) %>%
  ggplot() +
  geom_point(mapping = aes(x = bill_length_mm, y = flipper_length_mm, color = species),
             na.rm = TRUE,
             show.legend = FALSE) +
  scale_color_brewer(palette = "Set1") +
  labs(x = "bill length",
       y = "flipper length")

p3 <- filter(penguins, !is.na(sex)) %>%
  ggplot() +
  geom_point(mapping = aes(x = bill_depth_mm, y = flipper_length_mm, color = species),
             na.rm = TRUE,
             show.legend = FALSE) +
  scale_color_brewer(palette = "Set1") +
  labs(x = "bill depth",
       y = "flipper length")

p4 <- filter(penguins, !is.na(sex)) %>%
  ggplot() +
  geom_point(mapping = aes(x = flipper_length_mm, y = body_mass_g, color = species),
             na.rm = TRUE,
             show.legend  = TRUE) +
  scale_color_brewer(palette = "Set1") +
  labs(x = "flipper length",
       y = "body mass")
```

The plots of the features are as shown below

```
(p1 + p2) / (p3 + p4) + plot_layout(guides = 'collect') +
  plot_annotation(title = "Penguins Body Features",
                  subtitle = "Body features variation by species", theme = theme_grey())
```



Penguins Body Features

Body features variation by species

3. Classification Modelling using Linear Discriminant Analysis

The sections below creates a model to predict the sex of the penguins with bill depth, bill length, flipper length and body mass as the 4 predictors. The output response is a two level response variable having values as either male or a female.

The linear discriminant analysis (LDA) is the most common classification modelling technique that can be used for this.

The below code prepares the base penguins data after a bit of wrangling and munging and thereafter the data set is split into a 80:20 ratio of training set and test set respectively.

```
tbl_filtered_penguins <- filter(penguins, !is.na(sex) &
                                  !is.na(bill_length_mm) &
                                  !is.na(bill_depth_mm) &
                                  !is.na(flipper_length_mm) &
                                  !is.na(body_mass_g))

subset_index <- sample(x = 1:nrow(tbl_filtered_penguins), size = round(0.8 * nrow(tbl_filtere
d_penguins), digits = 0))


penguins_train <- tbl_filtered_penguins[subset_index, ]
penguins_test <- tbl_filtered_penguins[-subset_index, ]


# A linear model for the penguins data set predicting Sex
lda_fit <- lda(formula = sex ~ bill_length_mm + bill_depth_mm + flipper_length_mm + body_mass
_g, data = penguins_train)
```

So lda_fit is the classification model that has been built. The output response variable has been coded into the two levels as shown below

```
# How the qualitative predictor has been split across  factors
contrasts(penguins_train$sex)
```

```
##        male
## female    0
## male      1
```

We will now predic the response variable (i.e. the sex) based on the data of the training set. The below code does exactly that.

```
# Prediction Outputs

lda_pred <- predict(object = lda_fit, newdata = penguins_test)
```

The correlation matrix below will indicate how far the predicted value of the response variable matches/does not match with the actual values of the response variable.

```
# Correlation Matrix
table(lda_pred$class, penguins_test$sex)
```

```
##         
##          female male
##   female     29    1
##   male        2   35
```

The next set of code outputs the error rates and success rates of the model based on the training set

```
# success rate in prediction
mean(lda_pred$class == penguins_test$sex)
```

```
## [1] 0.9552239
```

```
# error rate in prediction
mean(lda_pred$class != penguins_test$sex)
```

```
## [1] 0.04477612
```

So as we can see the success rate of 0.96 of the model is quite high based on a 80:20 split of the data. The error rate of 0.04 likewise is also quite low indicating the model is predicting a response that matches with the actual data quite nicely.

3.1 Assessing Model Accuracy

The assessment of the model accuracy is a subject which figures prominently in every statistical modelling subject and there are many ways to assess the model accuracy.

We will study the model accuracy in the following ways

1. Method 1 :- Using random sampling and splitting the data set into 80:20 ratio 330 times (the total observations in the input data set)
2. Method 2:- Using Validation Set Approach and then assess the error rates of the predicted response
3. Method 3:- Using Leave-One-Out LOOV Cross Validation approach and then asses the error rates of the predicted response
4. Method 4:- Finally use the K-Fold Cross Validation Technique to study and plot the error rates by using k = to k = 50

3.1.1 Method 1. Random Sampling 330 times and using a 80:20 Data Split

We will next split the data 80:20 for around 330 times to see the average error and success rates of the model

```
set.seed(1234)



fn_populate_rates <- function(){

  v_success_rates <- vector(mode = "double", length = nrow(tbl_filtered_penguins))
  v_error_rates <- vector(mode = "double", length = nrow(tbl_filtered_penguins))


  for(i in seq(nrow(tbl_filtered_penguins))){
    subset_val <- sample(x = 1:nrow(tbl_filtered_penguins), size = round(0.8 * nrow(tbl_filte
red_penguins), digits = 0))
    lda_fit1 <- lda(formula = sex ~ bill_length_mm + bill_depth_mm + flipper_length_mm + body
_mass_g, data = tbl_filtered_penguins[subset_val, ])
    lda_pred1 <- predict(object = lda_fit1, newdata = tbl_filtered_penguins[-subset_val, ])
    v_success_rates[[i]] <- mean(lda_pred1$class == tbl_filtered_penguins[-subset_val, ]$sex)
    v_error_rates[[i]] <- mean(lda_pred1$class != tbl_filtered_penguins[-subset_val, ]$sex)
    i <- i + 1
  }

  df_rates <- cbind(v_success_rates, v_error_rates)
  return(df_rates)

}

df_rates_new <- as_tibble(fn_populate_rates())
```

```
mean(df_rates_new$v_success_rates)
```

```
## [1] 0.8925642
```

```
mean(df_rates_new$v_error_rates)
```

```
## [1] 0.1074358
```

As can be seen the mean error rate of 0.11 over this random sampling is quite low indicating the model is not performing badly and is predicting the sex of the penguins with a reasonable accuracy.

3.1.2 Method 2. Validation Set Approach

The Validation Set Approach involves splitting the data set into two equal halves and creating the model based on one half of the data set and using this model to predict and analyse the response on the other half.

The code set below creates a function to do this and then call this function to analyze the error and success rates on the test data.

```
set.seed(1234)


fn_validation_set <- function(){

  v_success_rates <- vector(mode = "double", length = 1)
  v_error_rates <- vector(mode = "double", length = 1)

  subset_val <- sample(x = 1:nrow(tbl_filtered_penguins), size = round(0.5 * nrow(tbl_filtered_penguins), digits = 0))
  lda_fit1 <- lda(formula = sex ~ bill_length_mm + bill_depth_mm + flipper_length_mm + body_mass_g, data = tbl_filtered_penguins[subset_val, ])
  lda_pred1 <- predict(object = lda_fit1, newdata = tbl_filtered_penguins[-subset_val, ])

  v_success_rates[[1]] <- mean(lda_pred1$class == tbl_filtered_penguins[-subset_val, ]$sex)
  v_error_rates[[1]] <- mean(lda_pred1$class != tbl_filtered_penguins[-subset_val, ]$sex)

  df_rates <- cbind(v_success_rates, v_error_rates)
  return(df_rates)

}

df_rates_cross_validation <- as_tibble(fn_validation_set())
```

The success and error rates of the Validation Set Approach is as below.

```
mean(df_rates_cross_validation$v_success_rates)
```

```
## [1] 0.9161677
```

```
mean(df_rates_cross_validation$v_error_rates)
```

```
## [1] 0.08383234
```

As can be seen from above the error rate of 0.08 in the validation set approach is also quite low indicating a good performance of the model on the training data set.

3.1.3 Method 3. Leave-one-out (LOOV) Cross Validation

This method entails using the entire set of observation as the training data and leaving aside one observation as the test data. This exercise is repeated for all the observations. The code snippet below does that.

```
set.seed(1234)



fn_loov_rates <- function(){

  v_success_rates <- vector(mode = "double", length = nrow(tbl_filtered_penguins))
  v_error_rates <- vector(mode = "double", length = nrow(tbl_filtered_penguins))


  for(i in seq(nrow(tbl_filtered_penguins))){

    lda_fit1 <- lda(formula = sex ~ bill_length_mm + bill_depth_mm + flipper_length_mm + body
_mass_g, data = tbl_filtered_penguins[-i, ])
    lda_pred1 <- predict(object = lda_fit1, newdata = tbl_filtered_penguins[i, ])
    v_success_rates[[i]] <- mean(lda_pred1$class == tbl_filtered_penguins[i, ]$sex)
    v_error_rates[[i]] <- mean(lda_pred1$class != tbl_filtered_penguins[i, ]$sex)
    i <- i + 1
  }

  df_rates <- cbind(v_success_rates, v_error_rates)
  return(df_rates)

}

df_loov_rates <- as_tibble(fn_loov_rates())
```

The error and success rates are as below

```
mean(df_loov_rates$v_success_rates)
```

```
## [1] 0.8948949
```

```
mean(df_loov_rates$v_error_rates)
```

```
## [1] 0.1051051
```

The test error rate of 0.11 with the LOOV Approach also has a low value again indicating a good performance of the model based on this sampling approach

3.1.4 Method 4. K-Fold Cross Validation Approach

The k-fold cross validation approach entails splitting the input data set into k groups and then creating the model on k-1 groups as the training set and applying the built-up model into the kth group and study the errror and success rates. Here we build an iterative loop on top of this entire approach and observe and visualize the error and success rates of each of the k-fold values ranging from k = 3 groups to k = 50 groups.

The code below does that.

```
# Generalizing to k = N groups
# and using this generalization to study the
# model result of k = 3 to k = 50 groups in the observation

set.seed(1234)

fn_kfold_rates_new <- function(k) {

  sub_df <- as.data.frame(x = 1:nrow(tbl_filtered_penguins))

  colnames(sub_df) <- "row_index_no"

  sub_df$bucket <- trunc(sub_df$row_index_no / round(nrow(sub_df)/k, digits = 0))


  v_success_rates <- vector(mode = "double", length = max(sub_df$bucket))
  v_error_rates <- vector(mode = "double", length = max(sub_df$bucket))


  for(i in seq(max(sub_df$bucket))){

    lda_fit1 <- lda(formula = sex ~ bill_length_mm + bill_depth_mm + flipper_length_mm + body
_mass_g, data = tbl_filtered_penguins[-sub_df[sub_df$bucket == i-1, ]$row_index_no

, ])
    lda_pred1 <- predict(object = lda_fit1, newdata = tbl_filtered_penguins[sub_df[sub_df$buc
ket == i-1, ]$row_index_no, ])
    v_success_rates[[i]] <- mean(lda_pred1$class == tbl_filtered_penguins[sub_df[sub_df$bucke
t == i-1, ]$row_index_no, ]$sex)
    v_error_rates[[i]] <- mean(lda_pred1$class != tbl_filtered_penguins[sub_df[sub_df$bucket
 == i-1, ]$row_index_no, ]$sex)
    i <- i + 1
  }


  df_rates <- data.frame(cbind(v_success_rates, v_error_rates))
  colnames(df_rates) <- c("success_rate", "error_rate")

  k_value <- rep(k, nrow(df_rates))

  df_rates <- cbind(df_rates, k_value)

  colnames(df_rates) <- c("success_rate", "error_rate", "k_value")
  return(df_rates)

}
```

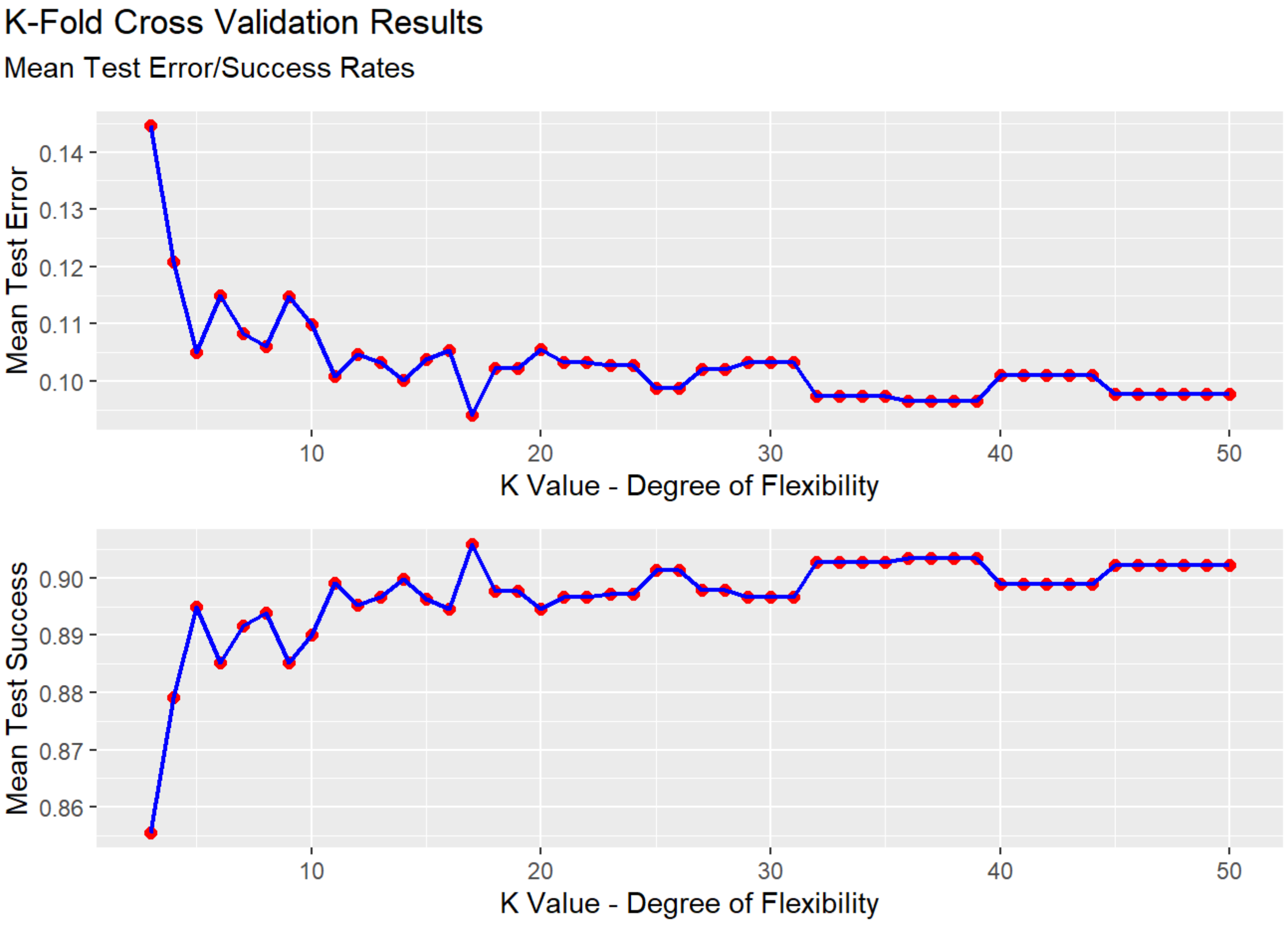
The above function creates the model, predicts the responses and holds out the error and success rates for an iteration of the k value. The below code snippet uses the map function of the purrr package to call the above function iteratively and stores the results of error and success rates against each of the k values

We now visualize the results of k-fold cross validation for the range of k values from k=3 to k=50 groups.

The code snippet below prepares the data for visualization

```
p_error_rate <-  tbl_kfold_rates %>% group_by(k_value) %>%
  summarize(mean_test_error = mean(error_rate)) %>%
  ungroup() %>%
  ggplot(mapping = aes(x = k_value, y = mean_test_error)) +
  geom_point(colour = "red", size = 2.0) +
  geom_line(colour = "blue", size = 0.75) +
  labs(x = "K Value - Degree of Flexibility",
       y = "Mean Test Error")

p_success_rate <-  tbl_kfold_rates %>% group_by(k_value) %>%
  summarize(mean_test_success = mean(success_rate)) %>%
  ungroup() %>%
  ggplot(mapping = aes(x = k_value, y = mean_test_success)) +
  geom_point(colour = "red", size = 2.0) +
  geom_line(colour = "blue", size = 0.75) +
  labs(x = "K Value - Degree of Flexibility",
       y = "Mean Test Success")
```

The plot below shows the variation of test error and success rates for each of the k-fold values ranging from 3 to 50.



The above graph is an interesting one. The graph indicates that the mean error rates on the test data gradually comes down and settles down to a value of nearly 0.10. So when this model performs on a reasonably large set of data we can conclude that the predicting response of sex of the penguins will be incorrect in 10% of the cases. So a success rate of 90% can be attributed to the predictions of this model based on the Linear Discriminant Analysis.