Stats Lab Bird

Joel Alder, Connor Charlton, Samir Hauser

2023-05-02

Data pre processing

We first load in the data and preprocess it.

We transform our response variable values to 0 and 1.

```
d.bird$sex_genetics[d.bird$sex_genetics == "M"] <- "m"
d.bird$sex_genetics[d.bird$sex_genetics == "F"] <- "f"
d.bird$sex_genetics[d.bird$sex_genetics == "m"] <- 0
d.bird$sex_genetics[d.bird$sex_genetics == "f"] <- 1
```

Next we define a new variable season, which indicates the time of the ringing. We do this because some of the variables depend on the time of year; for example the weight is different in summer and winter.

```
d.bird$season <- ifelse(d.bird$month >= 3 & d.bird$month <= 5, "spring",
                        ifelse(d.bird$month >= 6 & d.bird$month <= 8, "summer",
                               ifelse(d.bird$month >= 9 & d.bird$month <= 11,
                                      "autumn", "winter")))
d.bird$season <- factor(d.bird$season, levels = c("winter", "spring", "summer", "autumn"),
                        labels = c("0", "1", "2", "3"))
```

We have two variables for the feather length, P1 and P8. We merge them into one variable and where this variable takes the value of P8 if P8 has a value, and takes the value of P1 otherwise. There are only 32 instances for which P1 has a value and P8 does not.

```
if (sum(!is.na(d.bird$P1)) > 0) {
  d.bird$P8[is.na(d.bird$P8)] <- d.bird$P1[is.na(d.bird$P8)]
} else {
  print("NO impuatation possible as P1 is NA")
}
```

There are some duplicates in the dataset which we remove. Additionally, there is a possibility that one bird is captured more than once. If this is the case we only use the latest observation. Afterwards, we select only the adult birds and keep the ones for which we have the labels. In the end we select the columns we want to use for training. These are all the different morphological traits.

```
# Remove duplicates
d.bird <- d.bird[!duplicated(d.bird, fromLast = TRUE), ]

# check multiple captures
freq_table <- table(d.bird$ringnr)
```

```
freq_table <- freq_table[freq_table > 1]
freq_table <- sort(freq_table, decreasing = T)
double_caputre = as.data.frame(freq_table)
d.bird <- d.bird[!duplicated(d.bird$ringnr, fromLast = TRUE), ]

# select only the adult birds
df_adult <- d.bird[d.bird$Age>1,]

# select only rows with no NA in the response
df_adult <- df_adult[complete.cases(df_adult["sex_genetics"]),]

# select only the needed columns
cols <- c("season", "Age", "Wing", "P8", "Tarsus", "weight", "Fat", "Muscle",
          "Bill_length", "sex_genetics")
df_adult_sub = data.frame(df_adult[cols])

# chaning type of variables
df_adult_sub$Fat <- as.factor(df_adult_sub$Fat)
df_adult_sub$Muscle <- as.factor(df_adult_sub$Muscle)
df_adult_sub$sex_genetics <- as.factor(df_adult_sub$sex_genetics)
str(df_adult_sub)
```

```
## 'data.frame':    981 obs. of  10 variables:
##  $ season      : Factor w/ 4 levels "0","1","2","3": 3 3 2 3 2 2 2 2 3 3 ...
##  $ Age         : num  5 4 4 5 4 4 4 6 5 5 ...
##  $ Wing        : num  123 116 123 123 116 ...
##  $ P8          : num  98 90 97 94 90 96.5 89 89.5 89.5 95 ...
##  $ Tarsus      : num  23.4 22.7 22.9 23.4 22.5 24.2 22.6 23.2 22.8 22.7 ...
##  $ weight      : num  38 40 37.5 34 32.5 35 33 34 38 35.5 ...
##  $ Fat         : Factor w/ 7 levels "0","1","2","3",..: 3 NA 2 2 2 2 2 2 3 2 ...
##  $ Muscle      : Factor w/ 3 levels "1","2","3": 2 NA 2 2 3 1 2 3 1 1 ...
##  $ Bill_length : num  NA NA 10.8 14.1 NA NA NA NA NA NA ...
##  $ sex_genetics: Factor w/ 2 levels "0","1": 1 2 1 1 2 1 2 2 2 1 ...
```

We now train three different models on this data - a decision tree, a random forest and lastly a logistic regression model with missing values imputed via using MICE. We use the accuracy of our predictions as our evaluation metric.

Decision Tree

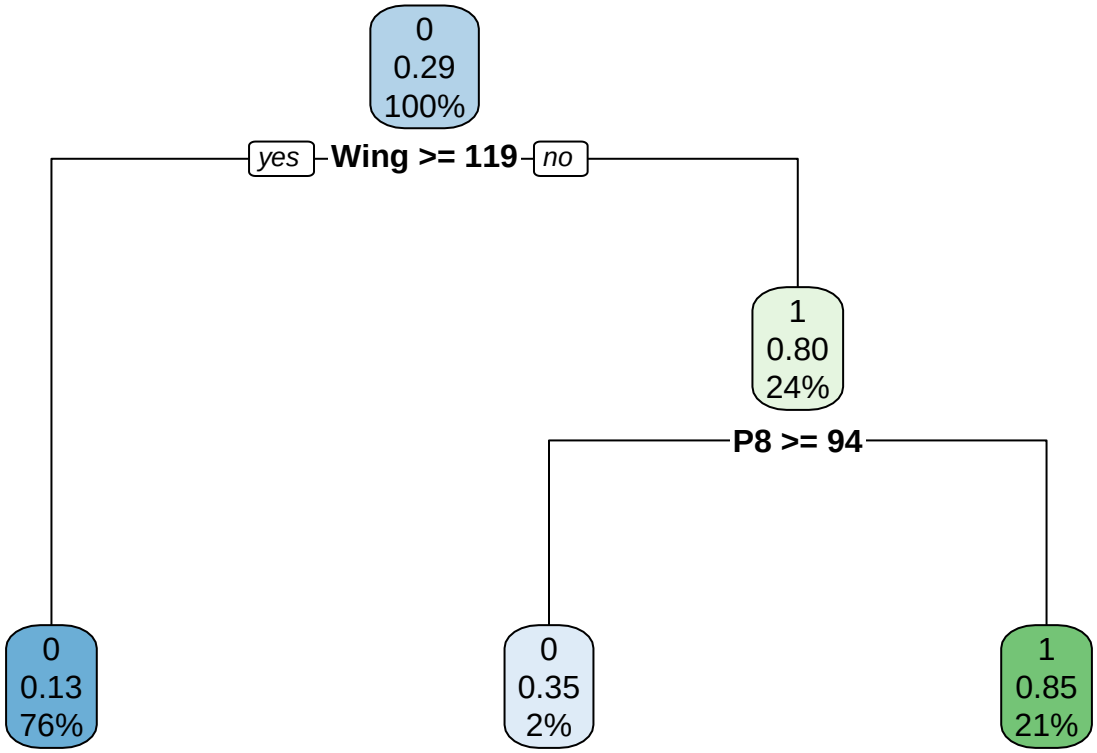
We use a ten fold cross-validation to get the accuracy score of the decision tree. The advantage of this method is that it can handle the missing values.

```
n <- dim(df_adult_sub)[1]
K <- 10
folds <- sample(cut(seq(1,n),breaks=K,labels=FALSE), replace = FALSE)
Fold.error <- numeric(K)
for (i in 1:K) {
  test.ind <- which(folds == i)
  tree_model.i <- rpart(sex_genetics ~ ., data = df_adult_sub[-test.ind,], method = "class")
  test_predictions.i <- predict(tree_model.i, df_adult_sub[test.ind,], type = "class")
  Fold.error[i] <- mean(test_predictions.i == df_adult_sub[test.ind,]$sex_genetics)
```

```
}
cat("Accuracy Decision Tree:", round(mean(Fold.error), digits = 5))
```

```
## Accuracy Decision Tree: 0.84714
```

We get an accuracy around **84.7 %**. One question that is raised is whether these two errors should be weighted differently. Maybe the misclassification of a female bird as a male bird should have a higher price than the opposite. We should keep in mind that the sex distribution at the population level is unbalanced, with two thirds being male. Now we plot the decision tree, to see which variables are important.



This produces a rather simple tree. The 0 values represent male birds and the 1 values represent female birds. As can be seen in the descriptive analysis of our data, the wing variable is the most informative for the prediction of genetic sex. There are only two variables in our tree.

Random Forest

We again use ten fold cross-validation to get the accuracy score of the random forest. The random forest can also handle missing values by specifying the na.action parameter. This will start by imputing the missing values using the column-wise median or mode. Then it grows a forest and computes proximities, before iterating through and constructing a new forest using these newly imputed values and repeating this process on repeat.

```
### Tuning of parameter done via this code
#ctrl <- trainControl(method = "cv", number = 10)
```

```r
#tunegrid <- expand.grid(.mtry=c(3:9), .ntree = seq(100, 1000, 100))
#ntree_tune <- train(sex_genetics ~ ., data = df_adult_sub, na.action = na.roughfix,
                     #method = customRF, tuneGrid = tunegrid, trControl = ctrl)

### 10 fold cross validation with function randomForest
Fold.error <- numeric(K)
for (i in 1:K) {
  test.ind <- which(folds == i)
  rf.i <- randomForest(sex_genetics ~ ., data = df_adult_sub[-test.ind,], ntree = 800,
                       mtry = 1, importance = TRUE, na.action = na.roughfix)
  test_data <- na.omit(df_adult_sub[test.ind,])
  test_predictions.i <- predict(rf.i, newdata = test_data, na.action = na.omit)
  Fold.error[i] <- mean(test_predictions.i == test_data$sex_genetics)
}
cat("Accuracy Random Forest:", round(mean(Fold.error), digits = 5))
```

```
## Accuracy Random Forest: 0.85275
```

We get an accuracy around **85.2 %**. We used the the train function to tune the two hyperparameter ntree and mtry. In the plot below we see the importance of the different variables. It yields that wing and P8 are the most important ones.

Logistic regression model

Logistic regression with complete cases

Before we use MICE to impute the missing values, we only use the complete cases with 741 observation. Because there might be some interactions we build a model which includes them. We do model selection with AIC and then we fit the best model with ten fold cross-validation to compute the accuracy score.

```
# use only the complete cases
df_adult_sub_comp <- na.omit(df_adult_sub)

# define the full model and fit it
fit <- glm(sex_genetics ~ season + Age + Wing + P8 + Tarsus + weight + Fat + Muscle
           + Bill_length + Wing*P8 + Wing*Bill_length + Wing*Tarsus + Wing*weight
           + P8*Bill_length + P8*Tarsus + P8*weight + Tarsus*weight + Tarsus*Bill_length
           + weight*Bill_length, data = df_adult_sub_comp, family = "binomial")

# use the step function to select the best model. It uses AIC criterion
step(fit, direction = "both", trace = 0)
```

```
##
## Call:  glm(formula = sex_genetics ~ season + Wing + P8 + Tarsus + Muscle +
##     Bill_length, family = "binomial", data = df_adult_sub_comp)
##
## Coefficients:
## (Intercept)      season1      season2         Wing           P8       Tarsus
##     77.8480       0.2269      -3.6361      -0.5272      -0.3373       0.3041
##     Muscle2      Muscle3  Bill_length
##      1.5315       2.7487       0.7169
##
## Degrees of Freedom: 740 Total (i.e. Null);  732 Residual
## Null Deviance:       885.3
## Residual Deviance: 464.1     AIC: 482.1
```

```
# we get this model as our best
best_model <- formula(sex_genetics ~ season + Wing + P8 + Tarsus + Muscle + Bill_length)

#do 10 fold cross validation with the best model
set.seed(123)
n <- dim(df_adult_sub_comp)[1]
folds <- sample(cut(seq(1,n),breaks=K,labels=FALSE), replace = FALSE)
Fold.error <- numeric(K)
for (i in 1:K) {
  test.ind <- which(folds == i)
  glm.i <- glm(best_model, data = df_adult_sub_comp[-test.ind,], family = binomial)
  prob_test_predictions.i <- predict(glm.i, df_adult_sub_comp[test.ind,], type = "response")
  test_predictions.i <- ifelse(prob_test_predictions.i > 0.5, 1, 0)
  Fold.error[i] <- mean(test_predictions.i == df_adult_sub_comp[test.ind,]$sex_genetics)
}
cat("Accuracy Logistic Model Complete Cases:", round(mean(Fold.error), digits = 5))
```

```
## Accuracy Logistic Model Complete Cases: 0.86099
```

This yields an accuracy of **86.1 %**. Now we look at the estimated coefficients of the model and there significance.

```
##                Estimate Std. Error    z value     Pr(>|z|)
## (Intercept) 77.8479749 7.27304118 10.7036346 9.786274e-27
## season1      0.2269047 0.25391932  0.8936094 3.715309e-01
## season2     -3.6361072 2.30854415 -1.5750651 1.152414e-01
## Wing        -0.5271909 0.06626136 -7.9562352 1.773531e-15
## P8          -0.3372846 0.06968261 -4.8402973 1.296450e-06
## Tarsus       0.3040828 0.16593143  1.8325811 6.686489e-02
## Muscle2      1.5315476 0.91143754  1.6803648 9.288636e-02
## Muscle3      2.7487494 1.02567619  2.6799387 7.363563e-03
## Bill_length  0.7169340 0.26025062  2.7547829 5.873109e-03
```

Logistic regression with imputing missing values

We again use ten fold cross-validation to compute the accuracy score of the logistic regression model. Because this model type can not handle missing values, the missing values are first imputed using MICE. This works by producing 35 separate imputations on which the logistic regression model is fitted independently, and the modal classification adopted. We use again the best model select by the AIC.

```
imp <- mice(df_adult_sub, seed = 123, print = F, m = 35, maxit = 10)
imp <- mice::complete(imp, "all")
Fold.error <- numeric(K)
for (i in 1:K) {
  test.ind <- which(folds == i)

  train_data <- df_adult_sub[-test.ind,]
  test_data <- df_adult_sub[test.ind,]

  # mice
  imp_train = mice(train_data, seed = 123, print = F, m=35, maxit = 10)

  # fit model
  fit_1 <- with(imp_train, glm(sex_genetics ~ season + Wing + P8 + Tarsus + Muscle
                               + Bill_length, family =  binomial))
  pooled <- pool(fit_1)

  # hack for predict
  pooled_lm = fit_1$analyses[[1]]
  pooled_lm$coefficients = summary(pooled)$estimate

  size = nrow(imp_train[[1]][test.ind,-(10)]) # remove sex

  # loop over imputed data
  dat = matrix(nrow = size, ncol = 35)
  for (k in 1:35)
  {
    predicted_values = predict(pooled_lm, newdata = imp[[k]][test.ind,-(10)],
                               type="response")
    binary_predictions <- ifelse(predicted_values > 0.5, 1, 0)
    dat[,k] = binary_predictions
  }
```

```
  #majority vote
  test_predictions.i <- apply(dat,1,Mode)

  Fold.error[i] <- mean(test_predictions.i == df_adult_sub[test.ind,]$sex_genetics)

}
cat("Accuracy Logistic Model with imputation:", round(mean(Fold.error), digits = 5))
```

```
## Accuracy Logistic Model with imputation: 0.87852
```

This results in a slightly improved accuracy of **87.9 %**. Now we examine the pooled estimates for the coefficients of the logistic regression model to the significance of them. We omit the fat variable because the standard error of the coefficient estimates was too high and the inclusion of the fat variable, as a predictor, explains too little variance in the regressand.

```
summary(pooled)
```

```
##          term    estimate  std.error  statistic       df      p.value
## 1 (Intercept) 83.77497783 7.02201982 11.9303249 751.0696 3.605425e-30
## 2     season1  0.08640196 0.22787975  0.3791559 843.0295 7.046676e-01
## 3     season2 -1.72071584 0.80862323 -2.1279575 714.2446 3.368243e-02
## 4        Wing -0.56643458 0.06287997 -9.0081885 869.7928 1.301949e-18
## 5          P8 -0.34252105 0.06481787 -5.2843617 794.3216 1.630851e-07
## 6      Tarsus  0.29398607 0.15829457  1.8572088 496.4544 6.387345e-02
## 7     Muscle2  0.93197754 0.55022727  1.6938047 890.7103 9.065202e-02
## 8     Muscle3  2.19493298 0.70279906  3.1231302 864.6421 1.848974e-03
## 9 Bill_length  0.72407727 0.24621127  2.9408779 418.5761 3.454424e-03
```

Similarly, to the decision tree this suggests that both the Wing and P8 variables are highly significant and the Bill length and Muscle variables are significant on a 5 % level.