Machine Learning Models on Abalone

Austin Ibarra

9/23/2020

#Abstract

Using the abalone dataset, I attempted to classify each observation as an adult or an infant based on the features of the data. To do this, I used three algorithms: random forest, ripper, and logistic regression for probability. I used the five main steps to organize my findings, with the first two being universal, and the last three steps split into a, b, and c to signify the three respective algorithms. Testing each of the model's fit and accuracy, it seems that the random forest model yields the highest accuracy of 83.51% while the ripper model falls slightly behind. I take high correlation into account when developing the final mmodel of logistic regression, ending up using a few predictor variables from the set.

#Step 1: Load in the dataset

I will be using the abalone dataset from the UC Irvine repository for the purpose of classification and logistic regeression. In order to do this, I recoded the variable "sex" into a binary variable called "Age" with factors adult (A) and infant (I). I also changed the variable names to easily differentiate between all of them.

Peering into the dataset we see we have several variables we can use in categorization or prediction. Though multicollinearity should be considered in the latter.

```
abalone <- read.csv("abalone_csv.csv")
names(abalone) <- c("Age", "Length", "Diameter", "Height", "Whole_Weight", "Shucked_Weight", "Viscera_W
head(abalone)
```

```
##   Age Length Diameter Height Whole_Weight Shucked_Weight Viscera_Weight
## 1   A  0.455    0.365  0.095       0.5140         0.2245         0.1010
## 2   A  0.350    0.265  0.090       0.2255         0.0995         0.0485
## 3   A  0.530    0.420  0.135       0.6770         0.2565         0.1415
## 4   A  0.440    0.365  0.125       0.5160         0.2155         0.1140
## 5   I  0.330    0.255  0.080       0.2050         0.0895         0.0395
## 6   I  0.425    0.300  0.095       0.3515         0.1410         0.0775
##   Shell_Weight Rings
## 1        0.150    15
## 2        0.070     7
## 3        0.210     9
## 4        0.155    10
## 5        0.055     7
## 6        0.120     8
```

#Step 2: Exploring and preparing the data

Checking the structure of the data

```
str(abalone)
```

```
## 'data.frame':    4177 obs. of  9 variables:
##  $ Age           : Factor w/ 2 levels "A","I": 1 1 1 1 2 2 1 1 1 1 ...
##  $ Length        : num  0.455 0.35 0.53 0.44 0.33 0.425 0.53 0.545 0.475 0.55 ...
##  $ Diameter      : num  0.365 0.265 0.42 0.365 0.255 0.3 0.415 0.425 0.37 0.44 ...
##  $ Height        : num  0.095 0.09 0.135 0.125 0.08 0.095 0.15 0.125 0.125 0.15 ...
##  $ Whole_Weight  : num  0.514 0.226 0.677 0.516 0.205 ...
##  $ Shucked_Weight: num  0.2245 0.0995 0.2565 0.2155 0.0895 ...
##  $ Viscera_Weight: num  0.101 0.0485 0.1415 0.114 0.0395 ...
```

```
##  $ Shell_Weight  : num  0.15 0.07 0.21 0.155 0.055 0.12 0.33 0.26 0.165 0.32 ...
##  $ Rings         : int  15 7 9 10 7 8 20 16 9 19 ...
```

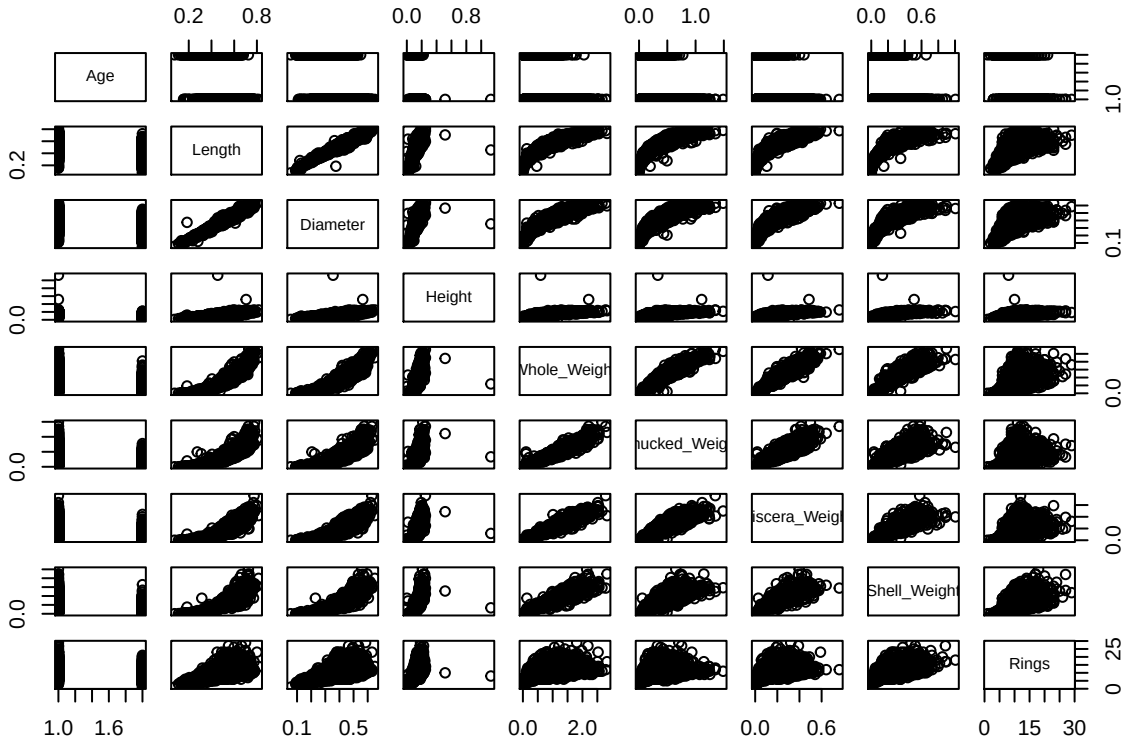
Summarize the data

```
summary(abalone)
```

```
##  Age          Length         Diameter          Height
##  A:2835   Min.   :0.075   Min.   :0.0550   Min.   :0.0000
##  I:1342   1st Qu.:0.450   1st Qu.:0.3500   1st Qu.:0.1150
##           Median :0.545   Median :0.4250   Median :0.1400
##           Mean   :0.524   Mean   :0.4079   Mean   :0.1395
##           3rd Qu.:0.615   3rd Qu.:0.4800   3rd Qu.:0.1650
##           Max.   :0.815   Max.   :0.6500   Max.   :1.1300
##   Whole_Weight    Shucked_Weight   Viscera_Weight    Shell_Weight
##  Min.   :0.0020   Min.   :0.0010   Min.   :0.0005   Min.   :0.0015
##  1st Qu.:0.4415   1st Qu.:0.1860   1st Qu.:0.0935   1st Qu.:0.1300
##  Median :0.7995   Median :0.3360   Median :0.1710   Median :0.2340
##  Mean   :0.8287   Mean   :0.3594   Mean   :0.1806   Mean   :0.2388
##  3rd Qu.:1.1530   3rd Qu.:0.5020   3rd Qu.:0.2530   3rd Qu.:0.3290
##  Max.   :2.8255   Max.   :1.4880   Max.   :0.7600   Max.   :1.0050
##      Rings
##  Min.   : 1.000
##  1st Qu.: 8.000
##  Median : 9.000
##  Mean   : 9.934
##  3rd Qu.:11.000
##  Max.   :29.000
```

Checking correlations for logistic regression. There appears to be high correlations among the explanatory variables such as the weights with eachother as well as the dimensions, so dropping some of these features in the future may prove useful for logistic regression.

```
pairs(abalone)
```



Split into training and test datasets

```
set.seed(126)
train <- sample(nrow(abalone), 0.7*nrow(abalone), replace = FALSE)
abalone_train <- abalone[train, ]
abalone_test <- abalone[-train, ]
```

#Step 3a: Training a random forest model on the data

Randomforest model with default parameters

```
library(randomForest)
abalone_rf <- randomForest(Age ~ ., data = abalone_train, importance = TRUE)
```

#Step 3b: Training a ripper model on the data

Ripper algorithm

```
library(RWeka)
abalone_rip <- OneR(Age ~ ., data = abalone_train)

abalone_pred <- predict(abalone_rip, abalone_test)
```

#Step 3c : Training a Logistic Regression model on the data

Logistic regression

```
abalone_glm <- glm(Age ~ ., family = "binomial", data = abalone_train)
```

#Step 4a: Evaluating the random forest model performance

Random forest accuracy

```
abalone_rf
```

```
##
## Call:
##  randomForest(formula = Age ~ ., data = abalone_train, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 16.97%
## Confusion matrix:
##      A   I class.error
## A 1771 210   0.1060071
## I  286 656   0.3036093
```

#Step 4b: Evaluating the ripper algorithm performance

Summary of ripper performance on training data

```
summary(abalone_rip)
```

```
##
## === Summary ===
##
## Correctly Classified Instances        2407               82.3469 %
## Incorrectly Classified Instances       516               17.6531 %
## Kappa statistic                          0.5922
## Mean absolute error                      0.1765
## Root mean squared error                  0.4202
## Relative absolute error                 40.4083 %
## Root relative squared error             89.9025 %
## Total Number of Instances             2923
##
## === Confusion Matrix ===
##
##     a    b   <-- classified as
##  1739  242 |    a = A
##   274  668 |    b = I
```

Evaluating the accuracy of the ripper algorithm on test data. The training model on the test dataset yields an accuracy of about 77%.

```
library(gmodels)
CrossTable(abalone_test$Age, abalone_pred,
           prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
           dnn = c('actual default', 'predicted default'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
```

```
## Total Observations in Table:  1254
##
##
##                 | predicted default
## actual default |         A |         I | Row Total |
## ---------------|-----------|-----------|-----------|
##              A |       733 |       121 |       854 |
##                |     0.585 |     0.096 |           |
## ---------------|-----------|-----------|-----------|
##              I |       169 |       231 |       400 |
##                |     0.135 |     0.184 |           |
## ---------------|-----------|-----------|-----------|
##   Column Total |       902 |       352 |      1254 |
## ---------------|-----------|-----------|-----------|
##
##
```

#Step 4c: Evaluating the logistic regression performance

Logistic regression parameter model.

```
summary(abalone_glm)
```

```
##
## Call:
## glm(formula = Age ~ ., family = "binomial", data = abalone_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0629  -0.5988  -0.2034   0.6781   4.0098
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.34994    0.44382   0.788 0.430421
## Length          16.16194    3.07826   5.250 1.52e-07 ***
## Diameter        -6.74111    3.82240  -1.764 0.077803 .
## Height          -4.40626    3.63767  -1.211 0.225786
## Whole_Weight    -7.03050    2.06229  -3.409 0.000652 ***
## Shucked_Weight   4.69540    2.37462   1.977 0.048005 *
## Viscera_Weight -10.62265    3.05842  -3.473 0.000514 ***
## Shell_Weight     3.53788    2.94996   1.199 0.230413
## Rings           -0.19186    0.02943  -6.519 7.07e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3674.6  on 2922  degrees of freedom
## Residual deviance: 2309.9  on 2914  degrees of freedom
## AIC: 2327.9
##
## Number of Fisher Scoring iterations: 6
```

#Step 5a: Improving the random forest model performance

Tuning and improving the random forest model. Using many different values for the number of trees and variables tried at each split, this tuned model yields the lowest error rate of 16.49%.

```
abalone_rf2 <- randomForest(Age ~ ., data = abalone_train, ntree = 650, mtry = 1, importance = TRUE)
abalone_rf2
```

```
##
## Call:
##  randomForest(formula = Age ~ ., data = abalone_train, ntree = 650,      mtry = 1, importance = TRUE
##                Type of random forest: classification
##                      Number of trees: 650
## No. of variables tried at each split: 1
##
##         OOB estimate of  error rate: 16.63%
## Confusion matrix:
##      A   I class.error
## A 1780 201   0.1014639
## I  285 657   0.3025478
```

#Step 5b: Improving the ripper performance

Step 2 contains a scatterplot matrix that shows heavy correlations with certain measurements when plotted against eachother. To reduce redundancy in the model, I decided to drop most of the predictors, except the ones that were significant according to the logistic regression output.

This did not improve the accuracy of the model, and the error rate remains capped at about 17%

```
abalone_rip1 <- OneR(Age ~ Length + Viscera_Weight + Rings + Whole_Weight, data = abalone_train)

summary(abalone_rip1)
```

```
##
## === Summary ===
##
## Correctly Classified Instances        2407               82.3469 %
## Incorrectly Classified Instances       516               17.6531 %
## Kappa statistic                          0.5922
## Mean absolute error                      0.1765
## Root mean squared error                  0.4202
## Relative absolute error                 40.4083 %
## Root relative squared error             89.9025 %
## Total Number of Instances             2923
##
## === Confusion Matrix ===
##
##     a    b   <-- classified as
##  1739  242 |    a = A
##   274  668 |    b = I
```

#Step 5c: Improving the logistic regression model performance

To reduce the redundancy and multicollinearity, I decided to run a backwards stepwise selection of the regression model.

The selection was semi-successful as I still have insignificant predictors in the model, however I check for variance inflation factors.

```
abalone_glm_step <- step(abalone_glm)
```

```
## Start:  AIC=2327.92
## Age ~ Length + Diameter + Height + Whole_Weight + Shucked_Weight +
```

```
##     Viscera_Weight + Shell_Weight + Rings
##
##                   Df Deviance    AIC
## - Shell_Weight     1   2311.3 2327.3
## <none>                 2309.9 2327.9
## - Height           1   2312.3 2328.3
## - Diameter         1   2313.1 2329.1
## - Shucked_Weight   1   2314.0 2330.0
## - Whole_Weight     1   2322.4 2338.4
## - Viscera_Weight   1   2322.6 2338.6
## - Length           1   2340.0 2356.0
## - Rings            1   2355.8 2371.8
##
## Step:  AIC=2327.35
## Age ~ Length + Diameter + Height + Whole_Weight + Shucked_Weight +
##     Viscera_Weight + Rings
##
##                   Df Deviance    AIC
## <none>                 2311.3 2327.3
## - Height           1   2313.5 2327.5
## - Shucked_Weight   1   2314.0 2328.0
## - Diameter         1   2314.0 2328.0
## - Viscera_Weight   1   2328.6 2342.6
## - Whole_Weight     1   2329.2 2343.2
## - Length           1   2340.8 2354.8
## - Rings            1   2356.4 2370.4
```

```
summary(abalone_glm_step)
```

```
##
## Call:
## glm(formula = Age ~ Length + Diameter + Height + Whole_Weight +
##     Shucked_Weight + Viscera_Weight + Rings, family = "binomial",
##     data = abalone_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0592  -0.5985  -0.1990   0.6799   3.9125
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)       0.25036    0.43365   0.577    0.564
## Length           15.91968    3.06416   5.195 2.04e-07 ***
## Diameter         -6.10523    3.77796  -1.616    0.106
## Height           -3.90162    3.44968  -1.131    0.258
## Whole_Weight     -5.10361    1.27497  -4.003 6.26e-05 ***
## Shucked_Weight    2.90906    1.82507   1.594    0.111
## Viscera_Weight -11.90877    2.88070  -4.134 3.57e-05 ***
## Rings            -0.18956    0.02928  -6.475 9.51e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3674.6  on 2922  degrees of freedom
```

```
## Residual deviance: 2311.4  on 2915  degrees of freedom
## AIC: 2327.4
##
## Number of Fisher Scoring iterations: 6
```

According to the variance inflation factors, there are high correlations with some of the predictors in the model.

```
library(faraway)
```

```
## Warning: package 'faraway' was built under R version 3.5.3
```

```
##
## Attaching package: 'faraway'
```

```
## The following object is masked from 'package:plyr':
##
##     ozone
```

```
round(vif(abalone_glm_step), 2)
```

```
##         Length       Diameter         Height   Whole_Weight Shucked_Weight
##         409.10         425.01          66.33        1171.65         490.49
## Viscera_Weight          Rings
##         299.53          27.33
```

Taking a closer look at the scatterplot matrix in step 2, it's easy to understand that predictors such as Viscera Weight and Whole_Weight would be highly correlated, so I drop those predictors in favor for my final logistic regression model.

```
abalone_reduce <- abalone %>% select(Age, Length, Viscera_Weight, Rings)
abalone_rglm <- glm(Age ~ ., family = "binomial", data = abalone_reduce)
summary(abalone_rglm)
```

```
##
## Call:
## glm(formula = Age ~ ., family = "binomial", data = abalone_reduce)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0510  -0.6100  -0.2171   0.6730   3.7959
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      1.30097    0.32485   4.005 6.21e-05 ***
## Length           7.43510    1.04499   7.115 1.12e-12 ***
## Viscera_Weight -25.04477    1.54456 -16.215  < 2e-16 ***
## Rings           -0.22544    0.02101 -10.728  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5244.9  on 4176  degrees of freedom
## Residual deviance: 3354.2  on 4173  degrees of freedom
## AIC: 3362.2
##
## Number of Fisher Scoring iterations: 6
```

#Conclusion

In an attempt to correctly classify the abalone dataset into adults and infants, I used three algorithms: Random Forest, Ripper, and logistic regression for prediction. Both classification algorithms, random forest and ripper, were both capped at an error rate slightly above 17%, but as we improved the model it became clear that the random forest yielded the smallest error rate at 16.49% while the ripper model was at 17.07%. Improving our logistic regression model, I checked the variance inflation factors and noted high correlations. To account for this, I dropped explanatory variables that were correlated with eachother and ended up only using length, viscera weight, and number of rings to predict the probability of the abalone being an adult or infant given the parameters of the model.

In future models, it may be of use to take note of the significant variation of abalone adult dimensions compared to infants. As infants, more or less, tend to have the same shell and weight dimensions, and grow up to have significantly different ones, resulting in a fan shaped pattern in the data. This may account for the capped error rate around 17% in both ripper and random forest models.