PSTAT 131 - HW 2

Ephets Head

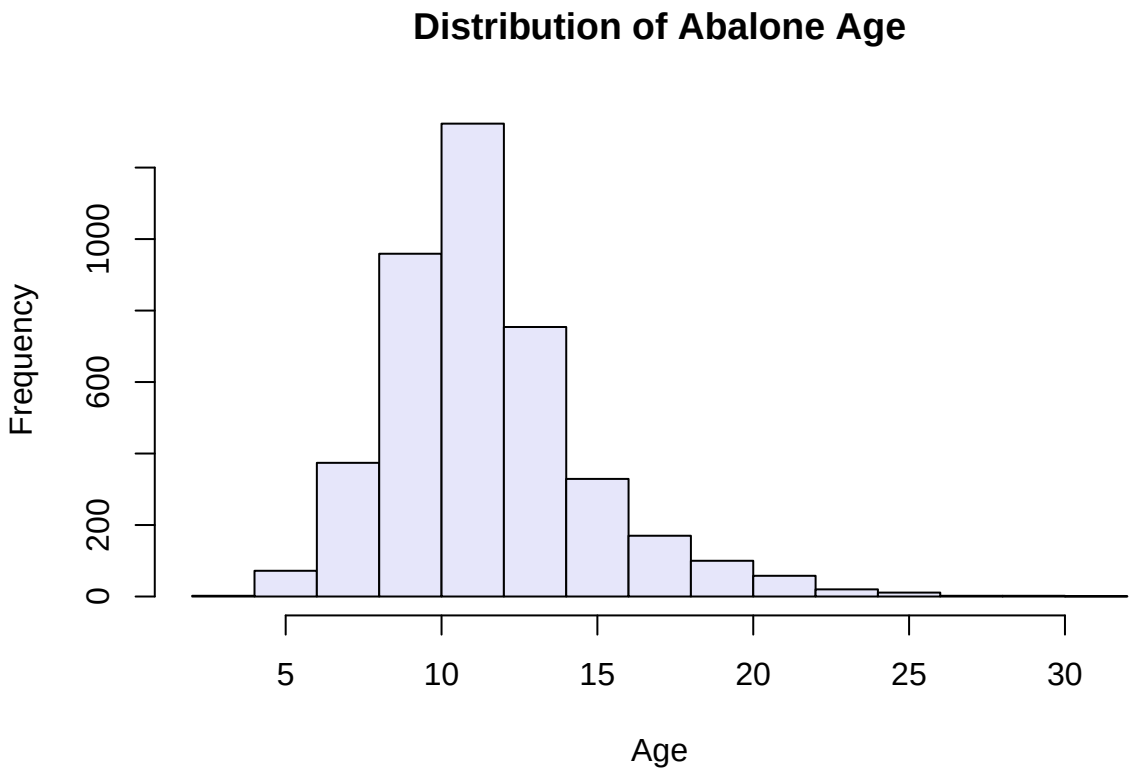
4/09/2022

First, we must read the *abalone* data set into R.

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

## -- Attaching packages --------------------------------------- tidymodels 0.2.0 --

## v broom        0.7.12     v rsample      0.1.1
## v dials        0.1.0      v tune         0.2.0
## v infer        1.0.0      v workflows    0.2.6
## v modeldata    0.1.1      v workflowsets 0.2.1
## v parsnip      0.2.1      v yardstick    0.0.9
## v recipes      0.2.0

## -- Conflicts ------------------------------------------ tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Learn how to get started at https://www.tidymodels.org/start/

## # A tibble: 4,177 x 9
##    type  longest_shell diameter height whole_weight shucked_weight
##    <chr>         <dbl>    <dbl>  <dbl>        <dbl>          <dbl>
##  1 M             0.455    0.365  0.095        0.514         0.224
##  2 M             0.35     0.265  0.09         0.226         0.0995
##  3 F             0.53     0.42   0.135        0.677         0.256
##  4 M             0.44     0.365  0.125        0.516         0.216
##  5 I             0.33     0.255  0.08         0.205         0.0895
##  6 I             0.425    0.3    0.095        0.352         0.141
##  7 F             0.53     0.415  0.15         0.778         0.237
##  8 F             0.545    0.425  0.125        0.768         0.294
##  9 M             0.475    0.37   0.125        0.509         0.216
## 10 F             0.55     0.44   0.15         0.894         0.314
## # ... with 4,167 more rows, and 3 more variables: viscera_weight <dbl>,
## #   shell_weight <dbl>, rings <dbl>
```

Question 1: Your goal is to predict abalone age, which is calculated as the number of rings + 1.5. Add *age* to the dataset. Then, assess and describe the distribution of *age*.

```
#below we create a new column "age" in the abalone data set, where age = rings + 1.5
abalone$age <- abalone$rings + 1.5

#to visualize the distribution of age, we can plot a histogram of the data from the age column
hist(abalone$age, xlab="Age", col= "lavender", main="Distribution of Abalone Age")
```



From the graph above, the abalone age appears to follow an almost normal distribution, peaking at around 11 and then gradually decreasing in frequency. The majority of the abalones seem to be in the age range of about 6 to 18, with very small numbers observed to be above the age of 20.

Question 2: Split the abalone data into a training set and a testing set. Use stratified sampling. You should decide on appropriate percentages for splitting the data.

```
set.seed(1026)

#the data has a good amount of observations (around 4000), so allocating 80% to the training set is app
abalone_split <- initial_split(abalone, prop=0.8,strata= age)
abalone_train <- training(abalone_split)
abalone_test <- testing(abalone_split)

dim(abalone_split)
```

```
##   analysis assessment          n          p
##       3340        837       4177         10
```

The training set *abalone_train* includes 80% of the data set observations (exactly 3340), while the testing set *abalone_test* is made up of the other 20% (837 observations).

Question 3: Using the training data, create a recipe predicting the outcome variable, *age*, with all other predictor variables except *rings*. Why should you not use *rings* to predict *age*?

Since the *rings* variable is used directly in the formula to calculate *age*, they are very strongly correlated and including it would mess up the relationship between predictors and response in the model.

Steps for your recipe:

1. Dummy code any categorical predictors.

2. Create interactions between *type* and *shucked_weight*, *longest_shell* and *diameter*, *shucked_weight* and *shell_weight*.

3. Center all predictors.

4. Scale all predictors.

```
abalone_recipe <-
  recipe(age ~ type + longest_shell + diameter + height + whole_weight + shucked_weight +
           viscera_weight + shell_weight, data = abalone_train) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_interact(~ starts_with("type"):shucked_weight) %>%
  step_interact(~ longest_shell:diameter) %>%
  step_interact(~ shucked_weight:shell_weight) %>%
  step_center(all_predictors()) %>%
  step_scale(all_predictors())

abalone_recipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor          8
##
## Operations:
##
## Dummy variables from all_nominal_predictors()
## Interactions with starts_with("type"):shucked_weight
## Interactions with longest_shell:diameter
## Interactions with shucked_weight:shell_weight
## Centering for all_predictors()
## Scaling for all_predictors()
```

Question 4: Create and store a linear regression object using the "lm" engine.

```
lm_model<-
  linear_reg() %>%
  set_engine("lm")
```

Question 5: Set up an empty workflow, add the model created in Question 4, and then add the recipe created in Question 3.

```
abalone_flow <-
  workflow() %>%
  add_model(lm_model) %>%
  add_recipe(abalone_recipe)

ab_lm_fit <- fit(abalone_flow, abalone_train)
```

Question 6: Use your fit() object to predict the age of a hypothetical female abalone with longest_shell = 0.50, diameter = 0.10, height = 0.30, whole_weight = 4, shucked_weight = 1, viscera_weight = 2, shell_weight = 1.

```
new_data <- data.frame(type="F",longest_shell=0.5,diameter=0.1,height=0.3,whole_weight=4,shucked_weight=
predict(ab_lm_fit,new_data[1,])
```

```
## # A tibble: 1 x 1
##   .pred
##   <dbl>
## 1  23.7
```

The predicted value of *age* for the given abalone is about 23.69.

Question 7: Now you want to assess your model's performance. To do this, use the *yardstick* package.

First, create a metric set that includes R^2, RMSE, and MAE. Then, use *predict()* and *bind_cols()* to create a tibble of your model's predicted values from the training data along with the actual observed ages. Finally, apply your metric set to the tibble, report the results, and interpret the R^2 value.

```
#create metrics set
ab_metrics_set <- metric_set(rsq, rmse, mae)

#create vector of predicted ages from training set
ab_response <- predict(ab_lm_fit,abalone_train)

#binds predicted ages column with column of actual observed ages from training set
ab_tibble <- bind_cols(ab_response, abalone_train %>% select(age))
head(ab_tibble, 10)
```

```
## # A tibble: 10 x 2
##    .pred   age
##    <dbl> <dbl>
##  1  9.39   8.5
##  2  8.11   8.5
##  3  9.38   9.5
##  4  9.80   8.5
##  5  9.99   9.5
##  6 10.9    9.5
##  7  6.30   6.5
##  8  5.92   5.5
##  9  8.56   8.5
## 10 11.3    9.5
```

As we can see from the first 10 rows of the tibble, our observed and predicted ages seem relatively close, though not incredibly accurate (most predicted values are too large).

```
#computes the R^2, RMSE, and MAE of the predicted vs. observed values
ab_metrics_set(ab_tibble, truth= age, estimate= .pred)
```

```
## # A tibble: 3 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.549
## 2 rmse    standard       2.16
## 3 mae     standard       1.55
```

The RMSE of our tibble is about 2.16, the MAE is about 1.55, and the R^2 is about 0.55. Since the R^2 value is low, this means only about 55% of the variance in the response is explained by the model, which is not a great sign for model accuracy.