PSTAT 231 -Homework 2

Code

Yuer Hao

Linear Regression

For this lab, we will be working with a data set from the UCI (University of California, Irvine) Machine Learning repository (see website here (http://archive.ics.uci.edu/ml/datasets/Abalone)). The full data set consists of $4,177$ observations of abalone in Tasmania. (Fun fact: Tasmania (https://en.wikipedia.org/wiki/Tasmania) supplies about 25% of the yearly world abalone harvest.)

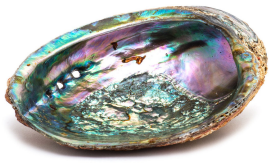


Fig 1. Inside of an abalone shell.

The age of an abalone is typically determined by cutting the shell open and counting the number of rings with a microscope. The purpose of this data set is to determine whether abalone age (**number of rings + 1.5**) can be accurately predicted using other, easier-to-obtain information about the abalone.

The full abalone data set is located in the `\data` subdirectory. Read it into *R* using `read_csv()`. Take a moment to read through the codebook (`abalone_codebook.txt`) and familiarize yourself with the variable definitions.

Make sure you load the `tidyverse` and `tidymodels`!

Hide

```
library(tidyverse)
library(ggplot2)
library(tidymodels)
library(corrplot)
library(ggthemes)
library(yardstick)
tidymodels_prefer()
set.seed(1000)
```

Hide

```
#load data set
setwd("/Users/Yuer_Hao/Desktop/PSTAT 131/PSTAT231 - HW2/PSTAT131 - homework-2/data")
abalone_data <- read.csv("abalone.csv")
head(abalone_data)
```

```
##   type longest_shell diameter height whole_weight shucked_weight viscera_weight
## 1    M         0.455    0.365  0.095       0.5140         0.2245         0.1010
## 2    M         0.350    0.265  0.090       0.2255         0.0995         0.0485
## 3    F         0.530    0.420  0.135       0.6770         0.2565         0.1415
## 4    M         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
```

Question 1

Your goal is to predict abalone age, which is calculated as the number of rings plus 1.5. Notice there currently is no `age` variable in the data set. Add `age` to the data set.

Assess and describe the distribution of `age`.

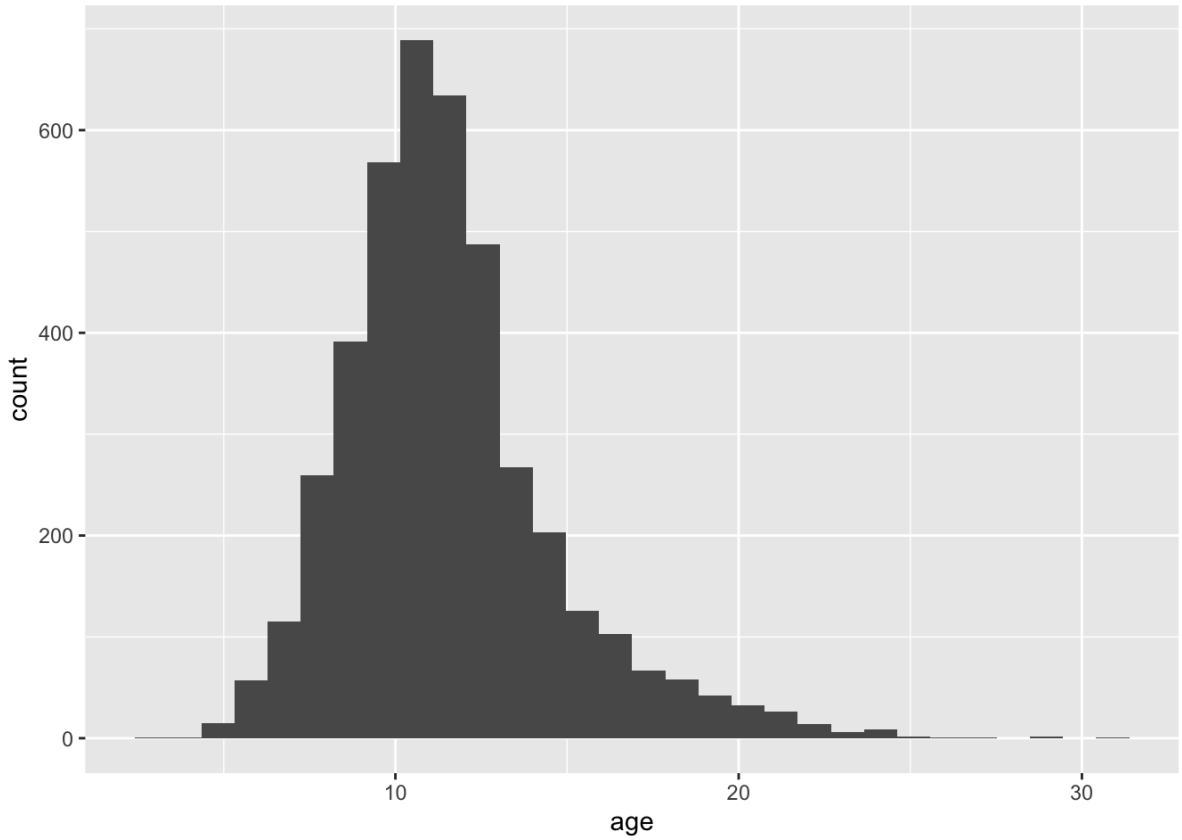
Hide

```
# Add age column to the abalone data set with "rings"+1.5
abalone <- abalone_data %>%
  mutate(abalone_data, age = rings + 1.5)
head(abalone)
```

```
##   type longest_shell diameter height whole_weight shucked_weight viscera_weight
## 1    M         0.455    0.365  0.095       0.5140         0.2245         0.1010
## 2    M         0.350    0.265  0.090       0.2255         0.0995         0.0485
## 3    F         0.530    0.420  0.135       0.6770         0.2565         0.1415
## 4    M         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  age
## 1        0.150    15 16.5
## 2        0.070     7  8.5
## 3        0.210     9 10.5
## 4        0.155    10 11.5
## 5        0.055     7  8.5
## 6        0.120     8  9.5
```

Hide

```
# Check the distribution of `age`
ggplot(data = abalone, aes(age)) +
  geom_histogram()
```



According to the plot, the distribution of age relatively follows the normal distribution with mean around 10-12. It is also slightly skewed to the right. Most of the age data falls between 5 and 18. However, there exists few of extreme outliers around age 30.

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. *Remember that you'll need to set a seed at the beginning of the document to reproduce your results.*

Hide

```
# Split the abalone data
abalone_split <- initial_split(abalone,prop = 0.80, strata = age)
abalone_train <- training(abalone_split)
abalone_test <- testing(abalone_split)
```

Question 3

Using the **training** data, create a recipe predicting the outcome variable, `age`, with all other predictor variables. Note that you should not include `rings` to predict `age`. Explain why you shouldn't use `rings` to predict `age`.

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, and
4. scale all predictors.

You'll need to investigate the `tidymodels` documentation to find the appropriate step functions to use.

Hide

```
#not include `rings` to predict `age`
aba_train_no_rings <- abalone_train %>%
  select(-rings)

#create a recipe
abalone_recipe <- recipe(age ~ ., data = aba_train_no_rings) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_interact(terms = ~ starts_with("type"):shucked_weight +
                  longest_shell:diameter +
                  shucked_weight:shell_weight) %>%
  step_center(all_predictors()) %>%
  step_scale(all_predictors())
```

Data rings cannot use to predict age since the age column is just the linear transformation (age = rings + 1.5) of the rings column. Therefore, they have exactly the same distribution and trend with shift.

Question 4

Create and store a linear regression object using the `"lm"` engine.

Hide

```
lm_model = linear_reg() %>%
  set_engine("lm")
```

Question 5

Now:

1. set up an empty workflow,
2. add the model you created in Question 4, and
3. add the recipe that you created in Question 3.

Hide

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

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.

Hide

```
lm_fit <- fit(lm_wkflow, aba_train_no_rings)
female_aba_pred <- data.frame(type = "F",
                              longest_shell = 0.50,
                              diameter = 0.10,
                              height = 0.30,
                              whole_weight = 4,
                              shucked_weight = 1,
                              viscera_weight = 2,
                              shell_weight = 1)

lm_fit %>%
  # This returns the parsnip object:
  extract_fit_parsnip() %>%
  # Now tidy the linear model object:
  tidy()
```

```
## # A tibble: 14 × 5
##    term                           estimate std.error statistic  p.value
##    <chr>                             <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)                      11.4      0.0373   307.    0
##  2 longest_shell                     0.312    0.290      1.08  2.81e- 1
##  3 diameter                          2.29     0.319      7.17  8.98e-13
##  4 height                            0.204    0.0699     2.92  3.52e- 3
##  5 whole_weight                      4.84     0.401     12.1   8.62e-33
##  6 shucked_weight                   -4.22     0.255    -16.5   5.29e-59
##  7 viscera_weight                   -0.928    0.160     -5.81  6.85e- 9
##  8 shell_weight                      1.76     0.219      8.04  1.21e-15
##  9 type_I                           -0.909    0.114     -7.98  2.01e-15
## 10 type_M                           -0.243    0.103     -2.36  1.82e- 2
## 11 type_I_x_shucked_weight           0.462    0.0857     5.39  7.55e- 8
## 12 type_M_x_shucked_weight           0.260    0.108      2.41  1.61e- 2
## 13 longest_shell_x_diameter         -2.78     0.399     -6.96  4.01e-12
## 14 shucked_weight_x_shell_weight    -0.164    0.206     -0.793 4.28e- 1
```

Hide

```
predict(lm_fit, new_data = female_aba_pred)
```

```
## # A tibble: 1 × 1
##   .pred
##   <dbl>
## 1  22.7
```

Question 7

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

1. Create a metric set that includes R^2, RMSE (root mean squared error), and MAE (mean absolute error).
2. 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 (these are needed to assess your model's performance).
3. Finally, apply your metric set to the tibble, report the results, and interpret the R^2 value.

Hide

```
#create a tibble
abalone_train_rlt <- predict(lm_fit, new_data = aba_train_no_rings %>% select(-age))
abalone_train_rlt <- bind_cols(abalone_train_rlt, aba_train_no_rings %>% select(age))

head(abalone_train_rlt)
```

```
## # A tibble: 6 × 2
##   .pred   age
##   <dbl> <dbl>
## 1  8.10   8.5
## 2  9.33   9.5
## 3 10.5    8.5
## 4 10.1    9.5
## 5 11.0    9.5
## 6  6.35   6.5
```

Hide

```
#create a metric set
abalone_metrics<-metric_set(rmse,rsq,mae)
abalone_metrics(abalone_train_rlt, truth=age,
                estimate=.pred)
```

```
## # A tibble: 3 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       2.15
## 2 rsq     standard       0.562
## 3 mae     standard       1.55
```

After applying metric set to the tibble, the results shows the value of R^2 value is 0.5618608 approximately which indicates that 56.18608% of the data fit the regression model.

Required for 231 Students

In lecture, we presented the general bias-variance tradeoff, which takes the form:

$$E[(y_0 - \hat{f}(x_0))^2] = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon)$$

where the underlying model $Y = f(X) + \epsilon$ satisfies the following:

- ϵ is a zero-mean random noise term and X is non-random (all randomness in Y comes from ϵ);
- (x_0, y_0) represents a test observation, independent of the training set, drawn from the same model;
- $\hat{f}(.)$ is the estimate of f obtained from the training set.

Question 8

Which term(s) in the bias-variance tradeoff above represent the reproducible error? Which term(s) represent the irreducible error?

- Reducible error: Bias and Variance $var(\hat{f}(x_0))$ and $[bias(\hat{f}(x_0))]^2$
- Irreducible error: zero-mean random noise $Var(\epsilon)$

Question 9

Using the bias-variance tradeoff above, demonstrate that the expected test error is always at least as large as the irreducible error.

Given that

$$E[(y_0 - \hat{f}(x_0))^2] = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon)$$

If we want the expected test error to stay as small as possible, the best way is to let the reducible error equal to 0, which means $\hat{f}(x_0)$ be unbiased and equals to $f(x_0)$. Then, $Var(\hat{f}(x_0)) = 0$ and $[Bias(\hat{f}(x_0))]^2 = 0$. But, since the irreducible error $Var(\epsilon)$ still exists, we have

$$E[(y_0 - \hat{f}(x_0))^2] = 0 + 0 + Var(\epsilon) = Var(\epsilon)$$

Thus, the expected test error is always at least as large as the irreducible error.

Question 10

Prove the bias-variance tradeoff.

Hints:

- use the definition of $Bias(\hat{f}(x_0)) = E[\hat{f}(x_0)] - f(x_0)$;
- reorganize terms in the expected test error by adding and subtracting $E[\hat{f}(x_0)]$

Proof:

$$\begin{aligned}
E[(y_0 - \hat{f}(x_0))^2] &= E[(f(x_0) + \epsilon - \hat{f}(x_0))^2] \\
&= E[(f(x_0) - \hat{f}(x_0))^2] + 2E[(f(x_0) - \hat{f}(x_0))\epsilon] + E[\epsilon^2] \\
&= E[(f(x_0) - \hat{f}(x_0))^2] + 2E[(f(x_0) - \hat{f}(x_0))\epsilon] + Var(\epsilon) \\
&= E[(f(x_0) - \hat{f}(x_0))^2] + 2E[(f(x_0) - \hat{f}(x_0))]E[\epsilon] + Var(\epsilon) \\
&= E[(f(x_0) - E[\hat{f}(x_0)]) - (\hat{f}(x_0) - E[\hat{f}(x_0)])^2] + Var(\epsilon) \\
&= E[(E[\hat{f}(x_0)] - f(x_0))^2] + E[(\hat{f}(x_0) - E[\hat{f}(x_0)]^2] - 2E[(f(x_0) - E[\hat{f}(x_0)])(\hat{f}(x_0) - E[\hat{f}(x_0)])] + Var(\epsilon) \\
&= [E[\hat{f}(x_0)] - f(x_0)]^2 + E[(\hat{f}(x_0) - E[\hat{f}(x_0)])^2] - 2(f(x_0) - E[\hat{f}(x_0)])E[(\hat{f}(x_0) - E[\hat{f}(x_0)])] + Var(\epsilon)
\end{aligned}$$

Based on the definition for bias($Bias(\hat{f}(x_0)) = E[\hat{f}(x_0)] - f(x_0)$)and variance, we can reorganize the terms:

$$\begin{aligned}
E[(y_0 - \hat{f}(x_0))^2] &= [Bias[\hat{f}(x_0)]]^2 + Var(\hat{f}(x_0)) - 2(f(x_0) - E[\hat{f}(x_0)])(E[\hat{f}(x_0)] - E[\hat{f}(x_0)]) + Var(\epsilon) \\
&= [Bias[\hat{f}(x_0)]]^2 + Var(\hat{f}(x_0)) + Var(\epsilon) \\
&= Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon).
\end{aligned}$$