131-Homework2

Caleb Mazariegos

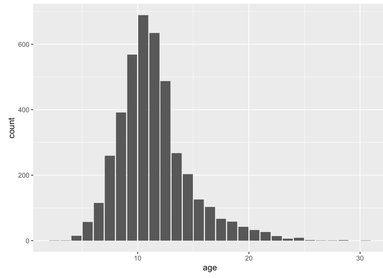
2022-04-10

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.

```
abalone <- abalone %>%
  mutate(age = rings + 1.5)

ggplot(abalone, aes(age)) + geom_bar()
```



Age has a normal distribution that is skewed right.

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.

```
set.seed(3435)

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.

```
abalone_recipe <- recipe(age ~ ., data = abalone_train %>% select(-rings)) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_interact(terms = ~starts_with("type"): shucked_weight)%>%
  step_interact(terms = ~ longest_shell:diameter) %>%
  step_interact(terms= ~ shucked_weight:shell_weight) %>%
step_normalize() %>% step_center(all_predictors()) %>%
step_scale()
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 and scaling for <none>
## Centering for all_predictors()
## Scaling for <none>
```

We should not use rings to predict age because since age is 1.5 + rings, the prediction will be perfect.

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

```
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.

```
lm_wflow <- 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.

```
lm_fit <- fit(lm_wflow, abalone_train)
lm_fit
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: linear_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 7 Recipe Steps
##
## • step_dummy()
## • step_interact()
## • step_interact()
## • step_interact()
## • step_normalize()
## • step_center()
## • step_scale()
##
## -- Model -----------------------------------------------------------------------
##
## Call:
## stats::lm(formula = ..y ~ ., data = data)
##
## Coefficients:
##                     (Intercept)                   longest_shell
##                         11.42335                         4.92120
##                        diameter                          height
##                        20.82298                         5.53388
##                    whole_weight                  shucked_weight
##                         8.73785                       -18.24370
##                  viscera_weight                    shell_weight
##                        -7.21225                        12.54605
##                          type_I                          type_M
##                        -2.00804                        -0.49584
##         type_I_x_shucked_weight         type_M_x_shucked_weight
##                         4.47388                         1.18334
##       longest_shell_x_diameter  shucked_weight_x_shell_weight
##                       -29.23234                       -0.02776
```

```
lm_fit %>%
  extract_fit_parsnip() %>%
  tidy()
```

```
## # A tibble: 14 × 5
##    term                        estimate std.error statistic  p.value
##    <chr>                          <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)                   11.4       0.0375  305.     0
##  2 longest_shell                  4.92      2.38      2.07   3.86e- 2
##  3 diameter                      20.8       3.15      6.61   4.59e-11
##  4 height                         5.53      1.63      3.39   7.10e- 4
##  5 whole_weight                   8.74      0.788    11.1    4.66e-28
##  6 shucked_weight               -18.2       1.12    -16.2    5.35e-57
##  7 viscera_weight                -7.21      1.44     -5.00   6.12e- 7
##  8 shell_weight                  12.5       1.53      8.20   3.32e-16
##  9 type_I                        -2.01      0.249    -8.07   9.36e-16
## 10 type_M                        -0.496     0.216    -2.29   2.21e- 2
## 11 type_I_x_shucked_weight        4.47      0.746     5.99   2.26e- 9
## 12 type_M_x_shucked_weight        1.18      0.442     2.68   7.41e- 3
## 13 longest_shell_x_diameter     -29.2       4.20     -6.95   4.32e-12
## 14 shucked_weight_x_shell_weight  -0.0278    1.72     -0.0161 9.87e- 1
```

```
random_data <- 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,
stringsAsFactors = TRUE)

prediction <- predict(lm_fit, new_data = random_data)
prediction
```

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

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 R2, 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 R2 value.

```
library("yardstick")

abalone_train_res <- predict(lm_fit, new_data = abalone_train %>% select(-age))
abalone_train_res %>%
  head()
```

```
## # A tibble: 6 × 1
##    .pred
##    <dbl>
## 1  8.03
## 2  9.68
## 3 10.4
## 4 10.1
## 5 10.9
## 6  6.26
```

```
abalone_train_res <- bind_cols(abalone_train_res, new_data = abalone_train %>% select(age))
abalone_train_res %>%
  head()
```

```
## # A tibble: 6 × 2
##    .pred   age
##    <dbl> <dbl>
## 1  8.03   8.5
## 2  9.68   8.5
## 3 10.4    8.5
## 4 10.1    9.5
## 5 10.9    9.5
## 6  6.26   6.5
```

```
rmse(abalone_train_res, truth = age, estimate = .pred)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        2.16
```

```
abalone_metrics <- metric_set(rmse, rsq, mae)
abalone_metrics(abalone_train_res, truth = age, estimate = .pred)
```

```
## # A tibble: 3 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       2.16
## 2 rsq     standard       0.551
## 3 mae     standard       1.55
```