

Homework 2

PSTAT 131/231

Contents

Linear Regression	1
-----------------------------	---

Linear Regression

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_data<- read.csv(file = 'abalone.csv')
age <- abalone_data$rings * 1.5
abalone_data <- cbind(abalone_data, age)

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  age
## 1         0.150    15 22.5
## 2         0.070     7 10.5
## 3         0.210     9 13.5
## 4         0.155    10 15.0
## 5         0.055     7 10.5
## 6         0.120     8 12.0
```

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.

```
# 2/3 of the data
set.seed(2784)

abalone_split <- initial_split(abalone_data, prop = 0.75, 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
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.

```
abalone_recipe <- recipe(age ~ type + longest_shell + diameter + height + whole_weight + shucked_weight
  step_dummy(all_nominal_predictors()) %>%
  step_center(all_predictors()) %>%
  step_scale(all_predictors()) %>%
  step_interact(terms = type ~ shucked_weight) %>%
  step_interact(terms = longest_shell ~ diameter) %>%
  step_interact(terms = shucked_weight ~ shell_weight)
```

```
abalone_recipe
```

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

Question 4

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

```
abalone_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.

```
abalone_workflow <- workflow() %>%  
  add_model(abalone_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`.

```
abalone_fit = fit(abalone_workflow, abalone_train)
```

```
## Warning: Interaction specification failed for: type ~ shucked_weight. No  
## interactions will be created.
```

```
predicted_df <- 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)
```

```
abalone_predict <- predict(abalone_fit, predicted_df)  
abalone_predict
```

```
## # A tibble: 1 x 1  
##   .pred  
##   <dbl>  
## 1  19.4
```

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.

```

abalone_metrics = metric_set(rsq, rmse, mae)
abalone_train_res <- predict(abalone_fit, new_data = abalone_train %>% select(-age))
abalone_train_res <- bind_cols (abalone_train_res, abalone_train %>%select(age))

abalone_metrics(abalone_train_res, truth = age, estimate = .pred)

```

```

## # A tibble: 3 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>       <dbl>
## 1 rsq     standard      0.541
## 2 rmse    standard      3.22
## 3 mae     standard      2.33

```

Required for 231 Students

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

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

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;
- $\hat{f}(\cdot)$ 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?

Question 9 Using the bias-variance tradeoff above, demonstrate that 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 $\text{Bias}(\hat{f}(x_0)) = E[\hat{f}(x_0)] - f(x_0)$;
- reorganize terms in the expected test error by adding and subtracting $E[\hat{f}(x_0)]$