Homework 4

PSTAT 131/231

Contents

Resampling

For this assignment, we will continue working with part of a Kaggle data set that was the subject of a machine learning competition and is often used for practicing ML models. The goal is classification; specifically, to predict which passengers would survive the Titanic shipwreck.

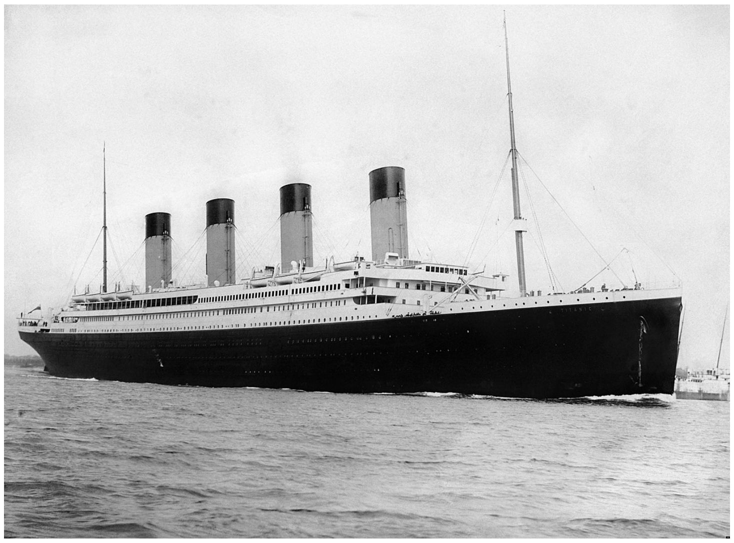

Figure 1: Fig. 1: RMS Titanic departing Southampton on April 10, 1912.

Load the data from `data/titanic.csv` into *R* and familiarize yourself with the variables it contains using the codebook (`data/titanic_codebook.txt`).

```
titanic_data <- read.csv("titanic.csv")

titanic_data$survived =  factor(titanic_data$survived, levels = c("Yes", "No"))
titanic_data$pclass =  factor(titanic_data$pclass)
```

Notice that `survived` and `pclass` should be changed to factors. When changing `survived` to a factor, you may want to reorder the factor so that *"Yes"* is the first level.

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

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

Create a recipe for this dataset **identical** to the recipe you used in Homework 3.

Question 1

Split the data, stratifying on the outcome variable, `survived`. You should choose the proportions to split the data into. Verify that the training and testing data sets have the appropriate number of observations.

```
set.seed(2000)

titanic_split <- initial_split(titanic_data, prop = 0.7, strata = survived )
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)

dim(titanic_train)
```

```
## [1] 623  12
```

```
dim(titanic_test)
```

```
## [1] 268  12
```

```
# hw3 same recipe
titanic_recipe <- recipe(survived ~ pclass + sex + age + sib_sp + parch + fare, data = titanic_train ) %
  step_impute_linear(age ) %>%
 step_dummy(all_nominal_predictors()) %>%
step_interact(terms = ~ starts_with("sex"):fare +
                  age:fare)
```

Question 2

Fold the **training** data. Use k-fold cross-validation, with $k = 10$.

```
titanic_folds <- vfold_cv(titanic_train, v = 10)
titanic_folds
```

```
## #  10-fold cross-validation
## # A tibble: 10 x 2
##    splits           id
##    <list>           <chr>
##  1 <split [560/63]> Fold01
##  2 <split [560/63]> Fold02
##  3 <split [560/63]> Fold03
##  4 <split [561/62]> Fold04
##  5 <split [561/62]> Fold05
##  6 <split [561/62]> Fold06
##  7 <split [561/62]> Fold07
##  8 <split [561/62]> Fold08
##  9 <split [561/62]> Fold09
## 10 <split [561/62]> Fold10
```

Question 3

In your own words, explain what we are doing in Question 2. What is k-fold cross-validation? Why should we use it, rather than simply fitting and testing models on the entire training set? If we **did** use the entire training set, what resampling method would that be?

We used K-fold cross validation. We split the data into 10 data sets. Since we have more data sets so even if the samples are limited, still we could have more models to fit in and then find the best model performed.

Question 4

Set up workflows for 3 models:

1. A logistic regression with the `glm` engine;
2. A linear discriminant analysis with the `MASS` engine;
3. A quadratic discriminant analysis with the `MASS` engine.

How many models, total, across all folds, will you be fitting to the data? To answer, think about how many folds there are, and how many models you'll fit to each fold.

```r
# logistic regression
log <- logistic_reg() %>%
  set_engine("glm") %>%
  set_mode("classification")

log_wkflow <- workflow() %>%
  add_model(log) %>%
  add_recipe(titanic_recipe)

# linear discriminant analysis
lda <- discrim_linear() %>%
  set_mode("classification") %>%
  set_engine("MASS")
lda_wkflow <- workflow() %>%
  add_model(lda) %>%
  add_recipe(titanic_recipe)

# quadratic discriminant analysis
qda <- discrim_quad() %>%
  set_mode("classification") %>%
  set_engine("MASS")
qda_wkflow <- workflow() %>%
  add_model(qda) %>%
  add_recipe(titanic_recipe)
```

Totally, we will have 30 models.

Question 5

Fit each of the models created in Question 4 to the folded data.

IMPORTANT: *Some models may take a while to run – anywhere from 3 to 10 minutes. You should NOT re-run these models each time you knit. Instead, run them once, using an R script, and store your results;*

look into the use of loading and saving. You should still include the code to run them when you knit, but set `eval = FALSE` *in the code chunks.*

```
log_fit = fit_resamples(log_wkflow, titanic_folds)
lda_fit = fit_resamples(lda_wkflow, titanic_folds)
qda_fit = fit_resamples(qda_wkflow, titanic_folds)
```

Question 6

Use `collect_metrics()` to print the mean and standard errors of the performance metric *accuracy* across all folds for each of the four models.

Decide which of the 3 fitted models has performed the best. Explain why. *(Note: You should consider both the mean accuracy and its standard error.)*

```
collect_metrics(log_fit)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.809    10  0.0168 Preprocessor1_Model1
## 2 roc_auc  binary     0.844    10  0.0166 Preprocessor1_Model1
```

```
collect_metrics(lda_fit)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.798    10  0.0208 Preprocessor1_Model1
## 2 roc_auc  binary     0.846    10  0.0166 Preprocessor1_Model1
```

```
collect_metrics(qda_fit)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.787    10  0.0176 Preprocessor1_Model1
## 2 roc_auc  binary     0.844    10  0.0151 Preprocessor1_Model1
```

We choose the log_fit which is logistic regression since we look inside the mean and the std error. It has the highest mean accuracy and also the smallest std error. Which is better than other two models.

Question 7

Now that you've chosen a model, fit your chosen model to the entire training dataset (not to the folds).

```
final_fit<- fit(log_wkflow, titanic_train)
```

Question 8

Finally, with your fitted model, use `predict()`, `bind_cols()`, and `accuracy()` to assess your model's performance on the testing data!

Compare your model's testing accuracy to its average accuracy across folds. Describe what you see.

```
perform_final <-
  predict(final_fit, titanic_test) %>%
  bind_cols(titanic_test %>% select(survived))%>%
  bind_cols(predict(final_fit, titanic_test, type = "prob")) %>%
  accuracy(truth = survived, .pred_class)

print(perform_final)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.799
```

```
collect_metrics(log_fit)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.809    10  0.0168 Preprocessor1_Model1
## 2 roc_auc  binary     0.844    10  0.0166 Preprocessor1_Model1
```

The accuracy of the model is around 80% and average across is also around 80%

Required for 231 Students

Consider the following intercept-only model, with $\epsilon \sim N(0, \sigma^2)$:

$$Y = \beta + \epsilon$$

where β is the parameter that we want to estimate. Suppose that we have n observations of the response, i.e. $y_1, ..., y_n$, with uncorrelated errors.

Question 9

Derive the least-squares estimate of β.

Question 10

Suppose that we perform leave-one-out cross-validation (LOOCV). Recall that, in LOOCV, we divide the data into n folds. What is the covariance between $\hat{\beta}^{(1)}$, or the least-squares estimator of β that we obtain by taking the first fold as a training set, and $\hat{\beta}^{(2)}$, the least-squares estimator of β that we obtain by taking the second fold as a training set?