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.

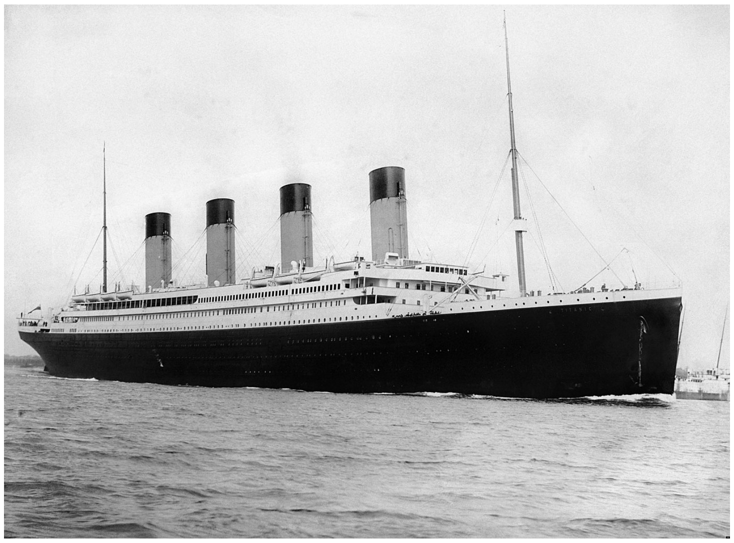

Figure 1: Fig. 1: RMS Titanic departing Southampton on April 10, 1912.

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

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.

```
library(tidymodels)
library(tidyverse)
```

```r
library(ISLR) # For the Smarket data set
library(ISLR2) # For the Bikeshare data set
library(discrim)
library(poissonreg)
library(corrr)
library(klaR) # for naive bayes
library(forcats)
library(corrplot)
library(pROC)
library(recipes)
library(rsample)
library(parsnip)
library(workflows)
tidymodels_prefer()
```

```r
titanic <- read.csv("titanic.csv") %>%
  mutate(survived = factor(survived,
                           levels = c("Yes", "No")),
         pclass = factor(pclass))
```

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.

```r
set.seed(2022)
titanic_split <- initial_split(titanic, prop = 0.80, strata = survived)
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)
dim(titanic_train)
```

```
## [1] 712  12
```

```r
dim(titanic_test)
```

```
## [1] 179  12
```

Question 2

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

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

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?
1. We using k-fold to divide the training data into 10 group.

2. k-fold cross-validation is an approach that involves randomly dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining k-1 folds.
3. We can avoid overfitting by using k-fold cross-validation.
4. Bootstrap.

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.

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

- logistic regression with the `glm` engine;

```
log_reg <- logistic_reg() %>%
  set_mode("classification")  %>%
  set_engine("glm")

log_wkflow <- workflow() %>%
  add_recipe(titanic_recipe) %>%
  add_model(log_reg)
```

- A linear discriminant analysis with the `MASS` engine;

```
lda_mod <- discrim_linear() %>%
  set_mode("classification") %>%
  set_engine("MASS")

lda_wkflow <- workflow() %>%
  add_model(lda_mod) %>%
  add_recipe(titanic_recipe)
```

- A quadratic discriminant analysis with the `MASS` engine.

```
qda_mod <- discrim_quad() %>%
  set_mode("classification") %>%
  set_engine("MASS")
```

```r
qda_wkflow <- workflow() %>%
  add_model(qda_mod) %>%
  add_recipe(titanic_recipe)
```

Since there have 3 different types of models and 10 folds for each type, then we will fit 30 models in total to the data.

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.*

```r
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.)*

```r
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     0.808    10  0.0190 Preprocessor1_Model1
## 2 roc_auc  binary     0.849    10  0.0164 Preprocessor1_Model1
```

```r
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     0.796    10  0.0225 Preprocessor1_Model1
## 2 roc_auc  binary     0.851    10  0.0168 Preprocessor1_Model1
```

```r
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     0.768    10  0.0158 Preprocessor1_Model1
## 2 roc_auc  binary     0.839    10  0.0175 Preprocessor1_Model1
```

The logistic fit is the best since it have highest mean and lowest ROC.

Question 7

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

```
log_fit_7 <- 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.

```
prediction <- predict(log_fit_7, new_data = titanic_test, type = "class") %>%
  bind_cols(titanic_test %>% select(survived)) %>%
  accuracy(truth = survived, estimate = .pred_class)
prediction
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.804
```

```
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     0.808    10  0.0190 Preprocessor1_Model1
## 2 roc_auc  binary     0.849    10  0.0164 Preprocessor1_Model1
```

Model's testing accuracy is 0.8044.
Average accuracy across folds is 0.807.
We can see that the model's testing accuracy is smaller than the model's average accuracy across folds.