Homework 4

PSTAT 131/231

Contents

Resampling 1
Required for 231 Students 4

```
library(tidymodels)
library(ISLR)
library(ISLR2)
library(tidyverse)
library(discrim)
library(poissonreg)
tidymodels_prefer()
```

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.

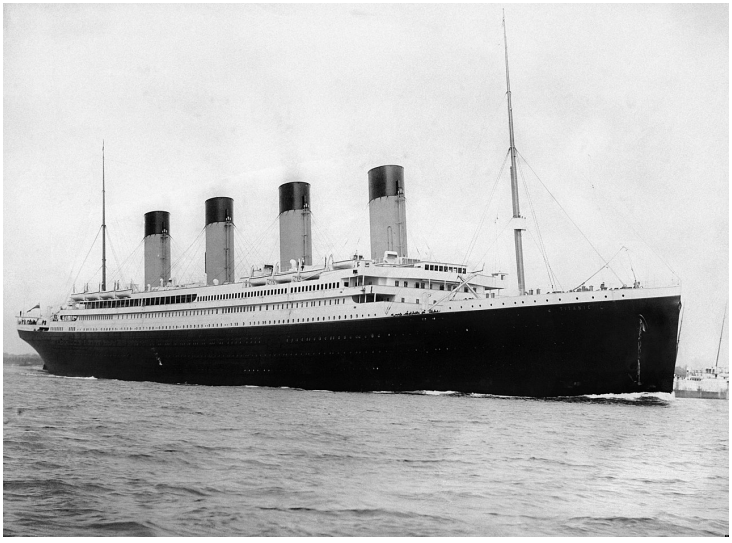

Figure 1: Fig. 1: RMS Titanic departing Southampton on April 10, 1912.

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`!

```
titanic <- read.csv('data/titanic.csv')

# preprocessing
titanic$survived <- factor(titanic$survived, levels = c("Yes","No"))
titanic$pclass <- as.factor(titanic$pclass)
```

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(42069) # for reproducibiility

# get training and testing data
titanic_split <- initial_split(titanic,prop=0.8,
                               strata=survived)

titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)

# recipe from hw 3
recipe_data<-titanic_train%>%select(c(survived,pclass,sex,age,sib_sp,parch,fare))
titanic_recipe <- recipe(survived ~ ., data = recipe_data)%>%
  step_impute_linear(age)%>%
  step_dummy(all_nominal_predictors())%>%
  step_interact(terms = ~starts_with("sex"):fare + age:fare)%>%
  step_center(all_predictors())%>%
  step_scale(all_predictors())
```

```
dim(titanic_train)
```

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

```
dim(titanic_test)
```

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

Question 2

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

```
titanic_fold <- vfold_cv(titanic_train)
```

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?

Answer: We are splitting the data into k roughly equal size clusters (folds). Then for each i in 1, 2, ..., k, we train the model on all folds except the i-th fold. Then we take the i-th fold to evaluate the accuracy of the model and record this statistic. Then we average across all iterations and take this average as our performance metric. The benefit is that this metric takes into account the models performance across the entire data set. If we were to just train on the entire training set, this would be 2-fold cross validation.

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.

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

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

# lda model
lda_mod <- discrim_linear() %>%
  set_mode("classification") %>%
  set_engine("MASS")

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

# qda model
qda_mod <- discrim_quad() %>%
  set_mode("classification") %>%
  set_engine("MASS")

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

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_rs <- log_wkflow %>% fit_resamples(titanic_fold)
lda_fit_rs <- lda_wkflow %>% fit_resamples(titanic_fold)
qda_fit_rs <- qda_wkflow %>% fit_resamples(titanic_fold)
```

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_rs)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.809    10  0.0180 Preprocessor1_Model1
## 2 roc_auc  binary     0.856    10  0.0217 Preprocessor1_Model1
```

```
collect_metrics(lda_fit_rs)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.792    10  0.0170 Preprocessor1_Model1
## 2 roc_auc  binary     0.856    10  0.0213 Preprocessor1_Model1
```

```
collect_metrics(qda_fit_rs)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.770    10  0.0238 Preprocessor1_Model1
## 2 roc_auc  binary     0.845    10  0.0248 Preprocessor1_Model1
```

We see that logistic regression yields the best result. Though LDA has almost identical performance.

Question 7

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

```
best_model_fit <- log_wkflow %>% fit(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.

```
log_test_acc <- predict(best_model_fit, new_data = titanic_test)%>%
  bind_cols(titanic_test%>%dplyr::select(survived))%>%
  accuracy(truth = survived, estimate = .pred_class)
log_test_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.821
```

I see that the accuracy on the testing data is actually slightly higher than the accuracy on the folds. However, overall pretty comparable accuracies.

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.

We want our estimate

$$\hat{\mathbf{Y}} = \mathbf{X}\boldsymbol{\beta} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \beta.$$

to satisfy that

$$\frac{1}{2}||\mathbf{Y} - \mathbf{X}\beta||_2^2 = \frac{1}{2}\sum_{i=1}^{n}(y_i - \beta)^2$$

is minimized. Taking the derivative with respect to β and setting equal to zero we get

$$\begin{aligned} \frac{d}{d\beta}\left(\frac{1}{2}\sum_{i=1}^{n}(y_i - \beta)^2\right) &= \sum_{i=1}^{n}(y_i - \beta) \\ &= -n\beta + \sum_{i=1}^{n} y_i \\ &\overset{set}{=} 0 \\ &\Rightarrow \beta = \frac{1}{n}\sum_{i=1}^{n} y_i \end{aligned}$$

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?

Answer: We have that

$$\hat{\beta}^{(1)} = \frac{1}{n-1}\sum_{i=2}^{n} y_i, \quad \hat{\beta}^{(2)} = \frac{1}{n-1}\sum_{i=1 i\neq 2}^{n} y_i$$

Therefore,

$$\begin{aligned} \mathrm{Cov}(\hat{\beta}^{(1)}, \hat{\beta}^{(2)}) &= \frac{1}{(n-1)^2}\mathrm{Cov}\left(\sum_{i=2}^{n} y_i, \sum_{i=1, i\neq 2}^{n} y_i\right) \\ &= \frac{1}{(n-1)^2}\sum_{i=3}^{n}\mathrm{Cov}(y_i, y_i) \\ &= \frac{(n-2)\sigma^2}{(n-1)^2} \end{aligned}$$

where all the cross-terms are zero by independence.