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.

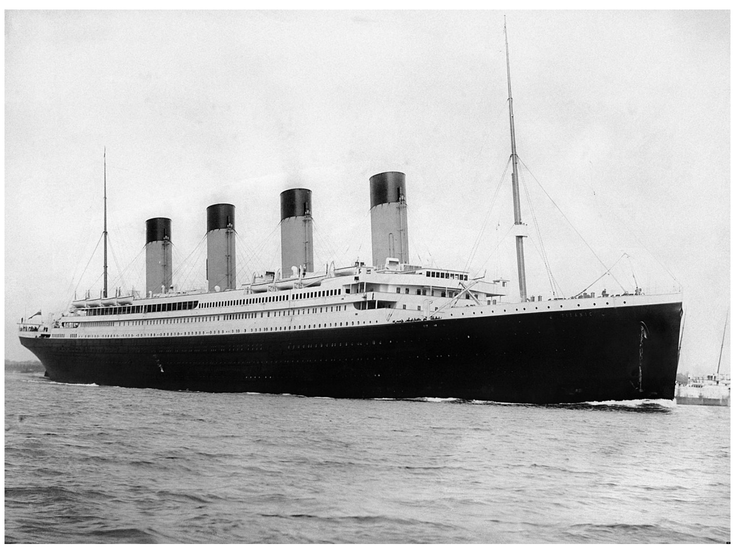

Figure 1: Fig. 1: RMS Titanic departing Southampton on April 10, 1912.

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.

```r
library(dplyr)
library(tidyverse)
library(tidymodels)
library(discrim)

titanic = read.csv("D:/UCSB/Spring 2022/PSTAT 131/PSTAT_131_HW/HW2/PSTAT-131/homework-4/homework-4/data/
titanic$survived = factor(titanic$survived, levels = c("Yes","No"))
titanic$pclass = factor(titanic$pclass)
titanic$sex = factor(titanic$sex)
head(titanic,6)
```

```
##   passenger_id survived pclass
## 1            1       No      3
## 2            2      Yes      1
## 3            3      Yes      3
## 4            4      Yes      1
## 5            5       No      3
## 6            6       No      3
##                                                  name    sex age sib_sp parch
## 1                             Braund, Mr. Owen Harris   male  22      1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38      1     0
## 3                              Heikkinen, Miss. Laina female  26      0     0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35      1     0
## 5                            Allen, Mr. William Henry   male  35      0     0
## 6                                    Moran, Mr. James   male  NA      0     0
##             ticket    fare cabin embarked
## 1        A/5 21171  7.2500  <NA>        S
## 2         PC 17599 71.2833   C85        C
## 3 STON/O2. 3101282  7.9250  <NA>        S
## 4           113803 53.1000  C123        S
## 5           373450  8.0500  <NA>        S
## 6           330877  8.4583  <NA>        Q
```

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
survived_split = initial_split(titanic, prop = 0.8, strata = survived)
survived_train = training(survived_split)
survived_test = testing(survived_split)

dim(survived_train)
```

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

```r
dim(survived_test)
```

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

```
# recipe
set.seed(826)

survived_recipe = recipe(survived ~ pclass + sex + age + sib_sp + parch + fare, data = survived_train) %>%
  step_impute_linear(age) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_interact(~ fare:starts_with("sex") + age:fare)

summary(survived_recipe)
```

```
## # A tibble: 7 x 4
##   variable type    role      source
##   <chr>    <chr>   <chr>     <chr>
## 1 pclass   nominal predictor original
## 2 sex      nominal predictor original
## 3 age      numeric predictor original
## 4 sib_sp   numeric predictor original
## 5 parch    numeric predictor original
## 6 fare     numeric predictor original
## 7 survived nominal outcome   original
```

Question 2

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

```
survived_fold = vfold_cv(survived_train, v=10)
survived_fold
```

```
## #  10-fold cross-validation
## # A tibble: 10 x 2
##    splits           id
##    <list>           <chr>
##  1 <split [640/72]> Fold01
##  2 <split [640/72]> Fold02
##  3 <split [641/71]> Fold03
##  4 <split [641/71]> Fold04
##  5 <split [641/71]> Fold05
##  6 <split [641/71]> Fold06
##  7 <split [641/71]> Fold07
##  8 <split [641/71]> Fold08
##  9 <split [641/71]> Fold09
## 10 <split [641/71]> 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?

Solution: We separate the training set into k exclusive subsets with equal sizes, then we hold out 1^{st} subset as the validation set and fit the model on the remaining $k-1$ subsets. After that, we get MSE in the first subset. Conclusively, we repeat this process k times for each subset to get different MSE, so that we can calculate their average MSE.

We use this method since we can well utilize the training set to get a reasonable MSE. Also, cross-validation deals with the variation of test MSE by showing the mean of all folds. Since the estimate of test MSE is highly variable and depends on training set, only fitting the entire training set may lead to an overestimate of the test MSE. This is called **A Validation Set Approach**.

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.

```
# a logistic regression
log_reg = logistic_reg() %>%
  set_engine("glm") %>%
  set_mode("classification")

log_wkflow = workflow() %>%
  add_model(log_reg) %>%
  add_recipe(survived_recipe)
```

```
# linear discriminant analysis
lda_mod <- discrim_linear() %>%
  set_mode("classification") %>%
  set_engine("MASS")

lda_wkflow = workflow() %>%
  add_model(lda_mod) %>%
  add_recipe(survived_recipe)
```

```
# quadratic discriminant analysis
qda_mod <- discrim_quad() %>%
  set_mode("classification") %>%
  set_engine("MASS")

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

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.

Solution:
For a single model, we have 10 folds and we fit once as each fold being a validation set. Since we have three models to train, our calculation should be $10 \times 3 = 30$.

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.*

```
set.seed(826)
control <- control_resamples(save_pred = TRUE)

log_res <- fit_resamples(log_wkflow, resamples = survived_fold, control = control)

lda_res <- fit_resamples(lda_wkflow, resamples = survived_fold, control = control)

qda_res <- fit_resamples(qda_wkflow, resamples = survived_fold, control = control)

save(log_res, lda_res, qda_res, file = "res.rda")
rm(log_res, lda_res, qda_res)
```

Question 6

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

```
load(file="res.rda")
# log_model
collect_metrics(log_res)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.815    10  0.0111 Preprocessor1_Model1
## 2 roc_auc  binary     0.845    10  0.0129 Preprocessor1_Model1
```

```
#lda_mod
collect_metrics(lda_res)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.806    10  0.0100 Preprocessor1_Model1
## 2 roc_auc  binary     0.846    10  0.0141 Preprocessor1_Model1
```

```
#qda_mod
collect_metrics(qda_res)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.792    10  0.0156 Preprocessor1_Model1
## 2 roc_auc  binary     0.844    10  0.0114 Preprocessor1_Model1
```

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.)*

Solution: Logistic regression has the largest accuracy 0.815. Since the standard deviation of each model is close and stands around 0.01, we can determine to choose logistic regression as the best fitted model.

Question 7

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

```
log_fit <- fit(log_wkflow, survived_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_acc <- predict(log_fit, new_data = survived_test, type = "class")%>%
  bind_cols(survived_test %>% dplyr::select(survived))%>%
  accuracy(truth = survived, estimate = .pred_class)

log_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.765
```

Solution:
We find that the accuracy of the one validation set approach is significantly smaller than the the average accuracy across folds. Therefore, it reflects the phenomenon that error may be overestimated by the one validation approach.

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?