Homework 4

PSTAT 131/231

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.

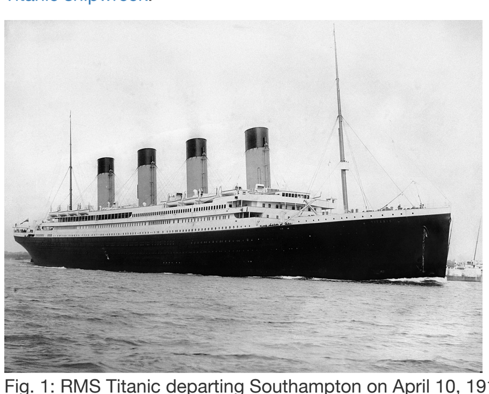

Fig. 1: RMS Titanic departing Southampton on April 10, 1912.

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`!

```r
library(ggplot2)
library(tidyverse)
library(tidymodels)
library(corrplot)
library(ggthemes)
library(corrr)
library(discrim)
#install.packages("pROC")
library(pROC)
library(klaR)
tidymodels_prefer()
 setwd("/Users/abhayzope/Desktop/Pstat 131")
Titanic_data=read.csv("titanic.csv")
Titanic_data$survived <- factor(Titanic_data$survived)
Titanic_data$pclass <- factor(Titanic_data$pclass)
Titanic_data %>%
  head()
```

```
##   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
```

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.

```r
set.seed(3435)

titanic_split <- initial_split(Titanic_data, prop = 0.80,
                                strata = survived)
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)

titanic_train %>%
  head()
```

```
##    passenger_id survived pclass                               name    sex age
## 1             1       No      3            Braund, Mr. Owen Harris   male  22
## 5             5       No      3           Allen, Mr. William Henry   male  35
## 7             7       No      1            McCarthy, Mr. Timothy J   male  54
## 8             8       No      3     Palsson, Master. Gosta Leonard   male   2
## 14           14       No      3        Andersson, Mr. Anders Johan   male  39
## 15           15       No      3 Vestrom, Miss. Hulda Amanda Adolfina female  14
##    sib_sp parch    ticket    fare cabin embarked
## 1       1     0 A/5 21171  7.2500  <NA>        S
## 5       0     0    373450  8.0500  <NA>        S
## 7       0     0     17463 51.8625   E46        S
## 8       3     1    349909 21.0750  <NA>        S
## 14      1     5    347082 31.2750  <NA>        S
## 15      0     0    350406  7.8542  <NA>        S
```

```r
dim(titanic_train)
```

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

```r
dim(titanic_test)
```

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

```r
titanic_recipe <- recipe(survived ~ pclass + sex + age + sib_sp + parch
                           + fare, data = titanic_train) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_impute_linear(age, impute_with = imp_vars(all_predictors())) %>%
  step_interact(terms = ~ sex:fare) %>%
  step_interact(terms = ~ age:fare)
```

Question 2

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

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

```
## #  10-fold cross-validation
## # A tibble: 10 × 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
```

```r
degree_grid <- grid_regular(degree(range = c(1, 10)), levels = 10)
degree_grid
```

```
## # A tibble: 10 × 1
##    degree
##     <dbl>
##  1      1
##  2      2
##  3      3
##  4      4
##  5      5
##  6      6
##  7      7
##  8      8
##  9      9
## 10     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?

K-fold cross validation is an alternative attempt to evaluate a model on some data. When performing k-fold cross validation, we are essentially dividing our data into folds and ensuring that each fold is used as a testing set at some point. K-fold cross validation ensures every observation from the original dataset has the chance of appearing in the training and test set. This is the key advantage that k-fold cross validation has over the validation set approach which we have been using up until now.

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.

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

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

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

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

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

qda_wkflow <- workflow() %>%
  add_model(qda_mod) %>%
  add_recipe(titanic_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.

We will be fitting thirty models to the data across all folds.

Question 5

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

```r
tune_res_logistic <- log_wkflow %>%
  fit_resamples(titanic_folds)



#tune_res_logistic <- tune_grid(
 # object = log_wkflow,
  #resamples = titanic_folds,
  #grid = degree_grid
#)
```

```r
tune_res_lda <- lda_wkflow %>%
        fit_resamples(titanic_folds)


#tune_res_lda <- tune_grid(
 # object = lda_wkflow,
#  resamples = titanic_folds,
  #grid = degree_grid
#)
```

```r
tune_res_qda <- qda_wkflow %>%
  fit_resamples(titanic_folds)

#tune_res_qda <- tune_grid(
 # object = qda_wkflow,
#  resamples = titanic_folds,
 # grid = degree_grid
#)
```

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.*

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.

```r
collect_metrics(tune_res_logistic)
```

```
## # A tibble: 2 × 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.811    10  0.0155 Preprocessor1_Model1
## 2 roc_auc  binary     0.849    10  0.0123 Preprocessor1_Model1
```

```r
collect_metrics(tune_res_lda)
```

```
## # A tibble: 2 × 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.794    10  0.0194 Preprocessor1_Model1
## 2 roc_auc  binary     0.849    10  0.0142 Preprocessor1_Model1
```

```r
collect_metrics(tune_res_qda)
```

```
## # A tibble: 2 × 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.791    10  0.0174 Preprocessor1_Model1
## 2 roc_auc  binary     0.841    10  0.0107 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.)*

The Logistic regression model is the best performing model as it has the highest mean and the lowest standard error.

Question 7

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

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

new_log_wkflow <- workflow() %>%
  add_model(new_log_reg) %>%
  add_recipe(titanic_recipe)

new_log_fit <- fit(new_log_wkflow, titanic_test)
```

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.

```r
log_modelpredict <- predict(new_log_fit, new_data = titanic_test, type = "prob")
log_modelaccuracy<- augment(new_log_fit, new_data = titanic_train) %>%
  accuracy(truth = survived, estimate = .pred_class)
bind_cols(log_modelpredict, log_modelaccuracy)
```

```
## # A tibble: 179 × 5
##    .pred_No .pred_Yes .metric  .estimator .estimate
##       <dbl>     <dbl> <chr>    <chr>          <dbl>
##  1   0.932     0.0682 accuracy binary         0.789
##  2   0.267     0.733  accuracy binary         0.789
##  3   0.150     0.850  accuracy binary         0.789
##  4   0.880     0.120  accuracy binary         0.789
##  5   0.273     0.727  accuracy binary         0.789
##  6   0.964     0.0357 accuracy binary         0.789
##  7   0.888     0.112  accuracy binary         0.789
##  8   0.500     0.500  accuracy binary         0.789
##  9   0.0543    0.946  accuracy binary         0.789
## 10   0.282     0.718  accuracy binary         0.789
## # … with 169 more rows
```

```r
#log_modelaccuracy
#log_modelpredict
```

We see a slight reduction on the model's testing accuracy in relation to its average accuracy across folds as the accuracy rate decreased from 80.4% to 78.9%. However, this is to be expected as most models generally perform slightly worse on testing data. Overall, while the logistic regression model is the most accurate model at our disposal, it still leaves a lot to be desired as approximately 20% of all predictions are incorrect.

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. $y_1, \ldots, y_n$, 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?