PSTAT 131 Homework 3

Luke Fields (8385924)

April 28, 2022

Below are the packages and libraries we are using in this assignment.

```r
library(corrplot)
library(discrim)
library(poissonreg)
library(corrr)
library(klaR) # for naive bayes
library(knitr)
library(MASS)
library(tidyverse)
library(tidymodels)
library(ggplot2)
library("dplyr")
library("yardstick")
tidymodels_prefer()
titanic <- read_csv("titanic.csv")
# set global chunk options: images will be 7x5 inches
knitr::opts_chunk$set(
    echo = TRUE,
    fig.height = 5,
    fig.width = 7,
    tidy = TRUE,
    tidy.opts = list(width.cutoff = 60)
)
opts_chunk$set(tidy.opts=list(width.cutoff=60),tidy=TRUE)
options(digits = 4)



## indents are for indenting r code as formatted text
## They may need to be adjusted depending on your OS
# if your output looks odd, increase or decrease indent
indent1 = '    '
indent2 = '        '
indent3 = '            '
```

Before we begin working with our model, we will factorize the survived and pclass variables first, making sure that "Yes" is the first level in our data set.

```r
set.seed(912) # can be any number
```

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

```
## Rows: 891 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): survived, name, sex, ticket, cabin, embarked
## dbl (6): passenger_id, pclass, age, sib_sp, parch, fare
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
titanic
```

```
## # A tibble: 891 x 12
##    passenger_id survived pclass name       sex     age sib_sp parch ticket  fare
##           <dbl> <fct>    <fct>  <chr>      <chr> <dbl>  <dbl> <dbl> <chr>  <dbl>
##  1            1 No       3      Braund, M~ male     22      1     0 A/5 2~  7.25
##  2            2 Yes      1      Cumings, ~ fema~    38      1     0 PC 17~ 71.3
##  3            3 Yes      3      Heikkinen~ fema~    26      0     0 STON/~  7.92
##  4            4 Yes      1      Futrelle,~ fema~    35      1     0 113803 53.1
##  5            5 No       3      Allen, Mr~ male     35      0     0 373450  8.05
##  6            6 No       3      Moran, Mr~ male     NA      0     0 330877  8.46
##  7            7 No       1      McCarthy,~ male     54      0     0 17463  51.9
##  8            8 No       3      Palsson, ~ male      2      3     1 349909 21.1
##  9            9 Yes      3      Johnson, ~ fema~    27      0     2 347742 11.1
## 10           10 Yes      2      Nasser, M~ fema~    14      1     0 237736 30.1
## # ... with 881 more rows, and 2 more variables: cabin <chr>, embarked <chr>
```

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(912)
titanic_split <- initial_split(titanic,
                               prop = 0.7, strata = survived)
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)
no_of_titanic_rows <- nrow(titanic)
no_of_train_rows <- nrow(titanic_train)
no_of_test_rows <- nrow(titanic_test)
no_of_missing_train <- colSums(is.na(titanic_train))
titanic_train
```

```
## # A tibble: 623 x 12
##    passenger_id survived pclass name      sex     age sib_sp parch ticket   fare
##           <dbl> <fct>    <fct>  <chr>     <chr> <dbl>  <dbl> <dbl> <chr>   <dbl>
##  1            1 No       3      Braund, ~ male     22      1     0 A/5 2~   7.25
##  2             5 No       3      Allen, M~ male     35      0     0 373450   8.05
##  3             8 No       3      Palsson,~ male      2      3     1 349909  21.1
##  4            13 No       3      Saunderc~ male     20      0     0 A/5. ~   8.05
##  5            19 No       3      Vander P~ fema~    31      1     0 345763  18
##  6            21 No       2      Fynney, ~ male     35      0     0 239865  26
##  7            25 No       3      Palsson,~ fema~     8      3     1 349909  21.1
##  8            27 No       3      Emir, Mr~ male     NA      0     0 2631     7.22
##  9            28 No       1      Fortune,~ male     19      3     2 19950  263
## 10            31 No       1      Uruchurt~ male     40      0     0 PC 17~  27.7
## # ... with 613 more rows, and 2 more variables: cabin <chr>, embarked <chr>
```

Above is what our training dataset looks like.

```
no_of_titanic_rows
```

```
## [1] 891
```

```
no_of_train_rows
```

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

```
no_of_test_rows
```

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

The previous three rows are the amount of observations in the titanic data set, as well as the train and test sets we created.

```
no_of_train_rows / no_of_titanic_rows
```

```
## [1] 0.6992
```

```
no_of_test_rows / no_of_titanic_rows
```

```
## [1] 0.3008
```

The above two rows give us the proportion of our training and test sets compared to our original titanic data set.

After performing a 70/30 train/test split, we see that there are 623 (69.9% of our original titanic data) and 268 (30.1% of our original titanic data) observations in the training data set and test datas et, respectively, so it is verified that the training and testing sets have the correct dimension.

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

titanic_recipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor          6
##
## Operations:
##
## Linear regression imputation for age
## Dummy variables from all_nominal_predictors()
## Interactions with sex_male:fare
## Interactions with age:fare
```

We are also creating our recipe above, which will be used for the rest of this model.

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
```

We see that there now 10 different folds of our split training data.

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?

In Question 2, we are splitting the data set into 10 different "folds", which will then all act as their own training and testing sets 10 different times. In my interpretation, k-fold cross validation is a way for us to continue to improve our model importance without ever touching the testing set, which is off limits. We use it instead of just simply fitting and testing models on the entire training set to allow our model's performance on the training data set to carry over to its performance on the testing set. If we used the entire training set, this would be a bootstrap re-sampling method.

Question 4: Set up workflows for 3 models: A logistic regression with the glm engine; A linear discriminant analysis with the MASS engine; 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.

We will first create a workflow for our k-fold for a logistic regression model.

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

titanic_log_workflow <- workflow() %>%
  add_model(titanic_log_reg) %>%
  add_recipe(titanic_recipe)
```

We will also create a workflow for our k-fold for a linear discriminant analysis model.

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

titanic_lda_workflow <- workflow() %>%
  add_model(titanic_lda_mod) %>%
  add_recipe(titanic_recipe)
```

We will finally create a workflow for our k-fold for a quadratic discriminant analysis model.

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

titanic_qda_workflow <- workflow() %>%
  add_model(titanic_lda_mod) %>%
  add_recipe(titanic_recipe)
```

There are 3 models with 10 folds each, so we will have to fit about 30 models in total for this project, which might take a while for our computers to compute.

Question 5: Fit each of the models created in Question 4 to the folded data.

First, we need to create a degree grid.

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

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

Then, we will apply the grid to our different models with the folds.

```
tune_titanic_log_reg <- tune_grid(
  object = titanic_log_workflow,
  resamples = titanic_folds,
  grid = titanic_degree_grid
)
```

```
## Warning: No tuning parameters have been detected, performance will be evaluated
## using the resamples with no tuning. Did you want to [tune()] parameters?
```

```
tune_titanic_lda <- tune_grid(
  object = titanic_lda_workflow,
  resamples = titanic_folds,
  grid = titanic_degree_grid
)
```

```
## Warning: No tuning parameters have been detected, performance will be evaluated
## using the resamples with no tuning. Did you want to [tune()] parameters?
```

```
tune_titanic_qda <- tune_grid(
  object = titanic_qda_workflow,
  resamples = titanic_folds,
  grid = titanic_degree_grid
)
```

```
## Warning: No tuning parameters have been detected, performance will be evaluated
## using the resamples with no tuning. Did you want to [tune()] parameters?
```

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.)

We will first analyze the metric

```
titanic_log_reg_met <- collect_metrics(tune_titanic_log_reg)[1,]
titanic_lda_met <- collect_metrics(tune_titanic_lda)[1,]
titanic_qda_met <- collect_metrics(tune_titanic_qda)[1,]
metric_collec <- rbind(titanic_log_reg_met, titanic_lda_met, titanic_qda_met)
Model_Type <- c("Log Reg", "LDA", "QDA")
metric_collec <- cbind(Model_Type, metric_collec)
metric_collec
```

```
##   Model_Type  .metric .estimator   mean  n std_err              .config
## 1    Log Reg accuracy     binary 0.8140 10 0.01637 Preprocessor1_Model1
## 2        LDA accuracy     binary 0.7963 10 0.01966 Preprocessor1_Model1
## 3        QDA accuracy     binary 0.7963 10 0.01966 Preprocessor1_Model1
```

Logistic regression performed the best in this performance metric because it had the smallest standard error (.0164) compared to both LDA and QDA (.01966 each).

Question 7: Now that you've chosen a model, fit your chosen model to the entire training dataset (not to the folds).

Now, we will fit our chosen model, logistic regression, to the entire training data set, and analyze its performance.

```
titanic_final_fit <- fit(titanic_log_workflow, titanic_train)
titanic_final_fit_acc <- augment(titanic_final_fit, new_data = titanic_train) %>%
  accuracy(truth = survived, estimate = .pred_class)
titanic_final_fit_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.822
```

Our model had a 0.822 accuracy estimate on the training data, which is pretty strong.

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.

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

```
## [1] 0.7985
```

The estimate for this model's accuracy in terms of performance on the testing data is 0.7985, or about 80% accuracy. This is about the same accuracy as both LDA and QDA in the folds accuracy, and slightly lower than the logistic regression accuracy in the fold modeling.

END OF HOMEWORK 4