PSTAT 131 HW 4

Luke Todd

4/23/2022

```
titanic$survived <- factor(titanic$survived, levels = c("Yes", "No"))
levels(titanic$survived)
```

Changing survived and pclass to factors

```
## [1] "Yes" "No"
```

```
titanic$pclass <- factor(titanic$pclass)
class(titanic$pclass)
```

```
## [1] "factor"
```

Question 1: Splitting the data

```
titanic_split <- initial_split(titanic, prop = 0.8,
                               strata = survived)

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

Creating a recipe from HW 3

```
titanic_recipe <- recipe(survived ~ pclass + sex + age +
                           sib_sp + parch + fare, data = titanic_train) %>%
  step_impute_linear(age) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_center(all_numeric_predictors()) %>%
  step_scale(all_numeric_predictors())
```

Question 2: Using k-fold cross-validation

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

Question 3: What is k-fold cross-validation?

K-fold cross-validation ensures that every observation from the original data set has the chance of appearing in training and test set. It "folds" the training set into your specified amount of folds (k), so that k iterations of modeling building and testing can be performed, eventually calculating metrics from each of the folds. Some of these metrics may include average, range, and standard deviation.

If we did just use the entire training set, this would be called the **validation set approach.**

Question 4: Set up workflows for 3 models

```r
# 1: Logistic regression

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

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

# 2: Linear Discriminant Analysis
lda_mod <- discrim_linear() %>%
  set_mode("classification") %>%
  set_engine("MASS")

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

# 3: Quadratic Discriminant Analysis
qda_mod <- discrim_quad() %>%
  set_mode("classification") %>%
  set_engine("MASS")

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

There are 10 folds and we will be fitting 3 models to each fold, so we will end up fitting 30 models total across all folds.

Question 5: Fit each of the models to the folded data

```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: collect_metrics()

```
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     0.804    10  0.0166 Preprocessor1_Model1
## 2 roc_auc  binary     0.858    10  0.0141 Preprocessor1_Model1
```

```
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     0.797    10  0.0182 Preprocessor1_Model1
## 2 roc_auc  binary     0.859    10  0.0137 Preprocessor1_Model1
```

```
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     0.792    10  0.0144 Preprocessor1_Model1
## 2 roc_auc  binary     0.850    10  0.0131 Preprocessor1_Model1
```

It can be seen that the log fit was the best model. It has the highest accuracy, as well as the lowest standard error when compared to the other two models.

Question 7: fit to entire training dataset (not folds)

```
final_log_fit <- fit(log_wkflow, data = titanic_train)
```

Question 8: assess model's performance

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

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.816
```

The testing data resulted in an accuracy of 0.8156, which is higher than the accuracy of 0.8048 that we saw on the training data. This shows that our data fits the model well.