Lab 14 - Cross Validation and Bootstrap Confidence Intervals

TENSORFLOW 2.0

12/10/2020

Load packages

```r
library(tidyverse)
library(tidymodels)
library(dsbox)
```

Exercise 1

```r
gss16_advfront <- gss16 %>%
  mutate(
    # Modify advfront to have fewer levels
    advfront = case_when(
      is.na(advfront) ~ advfront,
      advfront %in% c("Strongly agree", "Agree") ~ "Agree",
      TRUE ~ "Not agree"
    ),
    # Convert to factor with specified level order
    advfront = fct_relevel(advfront, "Agree", "Not agree"),
    # Modify polviews to have fewer levels
    polviews = case_when(
      str_detect(polviews, "[Ll]iberal") ~ "Liberal",
      str_detect(polviews, "[Cc]onservative") ~ "Conservative",
      TRUE ~ polviews
    ),
    # Convert to factor with specified level order
    polviews = fct_relevel(polviews, "Conservative", "Moderate", "Liberal"),
  ) %>%
  # Select the 4 variables of interest and drop rows with NA values
  select(advfront, educ, polviews, wrkstat) %>%
  drop_na()
```

Exercise 2

```r
set.seed(2222)
gss16_split <- initial_split(gss16_advfront)
gss16_train <- training(gss16_split)
gss16_test  <- testing(gss16_split)
```

Exercise 3

```
gss16_rec_1 <- recipe(advfront ~ polviews + wrkstat + educ, data = gss16_train) %>%
  step_other(wrkstat, threshold = 0.10, other = "Other") %>%
  step_dummy(all_nominal(), -all_outcomes())
```

Exercise 4

```
gss16_spec <- logistic_reg() %>%
  set_engine("glm")
```

Exercise 5

```
gss16_wflow_1 <- workflow() %>%
  add_model(gss16_spec) %>%
  add_recipe(gss16_rec_1)
```

Exercise 6

Now, lets find the efficiency of our first model on our training data.

```
set.seed(2222)
gss16_folds <- vfold_cv(gss16_train, v = 5)
gss16_fit_rs_1 <- gss16_wflow_1 %>%
  fit_resamples(gss16_folds)
```

```
##
## Attaching package: 'rlang'

## The following objects are masked from 'package:purrr':
##
##     %@%, as_function, flatten, flatten_chr, flatten_dbl, flatten_int,
##     flatten_lgl, flatten_raw, invoke, list_along, modify, prepend,
##     splice

##
## Attaching package: 'vctrs'

## The following object is masked from 'package:dplyr':
##
##     data_frame

## The following object is masked from 'package:tibble':
##
##     data_frame
```

```
collect_metrics(gss16_fit_rs_1, summarize = FALSE)
```

```
## # A tibble: 10 x 5
##    id    .metric  .estimator .estimate .config
##    <chr> <chr>    <chr>          <dbl> <chr>
##  1 Fold1 accuracy binary         0.81  Preprocessor1_Model1
##  2 Fold1 roc_auc  binary         0.659 Preprocessor1_Model1
##  3 Fold2 accuracy binary         0.874 Preprocessor1_Model1
##  4 Fold2 roc_auc  binary         0.563 Preprocessor1_Model1
##  5 Fold3 accuracy binary         0.814 Preprocessor1_Model1
##  6 Fold3 roc_auc  binary         0.558 Preprocessor1_Model1
##  7 Fold4 accuracy binary         0.834 Preprocessor1_Model1
##  8 Fold4 roc_auc  binary         0.648 Preprocessor1_Model1
##  9 Fold5 accuracy binary         0.874 Preprocessor1_Model1
## 10 Fold5 roc_auc  binary         0.611 Preprocessor1_Model1
```

```
collect_metrics(gss16_fit_rs_1)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.841     5  0.0141 Preprocessor1_Model1
## 2 roc_auc  binary     0.608     5  0.0209 Preprocessor1_Model1
```

The accuracy of our first model across the folds of the training data varied between 0.810 and 0.874, with the range of 0.064. Similarly, the area under the ROC curve of our first model for different folds of the training data varied between 0.558 and 0.659, the range of 0.101. The values in the accuracy and area under the ROC curve for different folds are consistent(similar to each other) and their ranges are relatively low. This suggests that the partition of training data created by splitting into 5 folds were similar to each other and did not have significant biases in them.

The average accuracy and average area under the ROC curve of our first model for different folds of the training data is 0.841 and 0.608 respectively.

Exercise 7

Now, lets find the efficiency of our second model on our training data.

```
gss16_rec_2 <- recipe(advfront ~ polviews + educ, data = gss16_train) %>%
  step_dummy(all_nominal(), -all_outcomes())

gss16_wflow_2 <- workflow() %>%
  add_model(gss16_spec) %>%
  add_recipe(gss16_rec_2)

set.seed(2222)
gss16_folds <- vfold_cv(gss16_train, v = 5)
gss16_fit_rs_2 <- gss16_wflow_2 %>%
  fit_resamples(gss16_folds)
collect_metrics(gss16_fit_rs_2)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.839     5  0.0140 Preprocessor1_Model1
## 2 roc_auc  binary     0.632     5  0.0149 Preprocessor1_Model1
```

The average accuracy and average area under the ROC curve of our first model for different folds of the training data is 0.839 and 0.632 respectively.

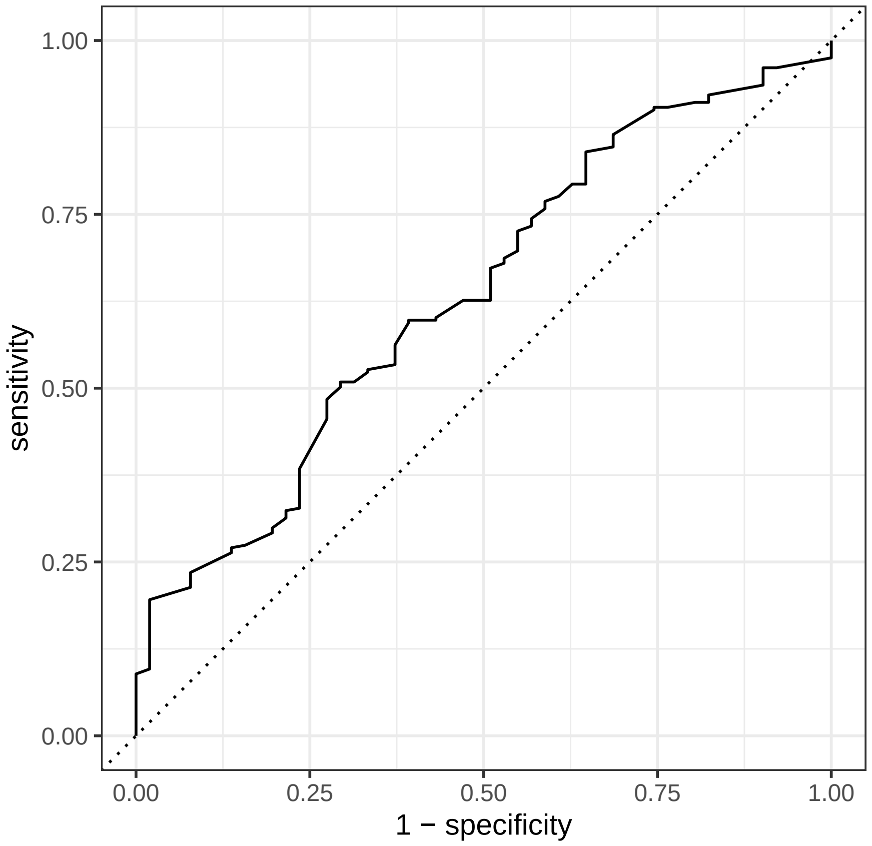
Exercise 8

The area under the ROC curve for our second model is higher than that for our first model. Hence we conclude that our second model performed better on our training data than the first model.

Exercise 9

Now, lets make predictions on our testing data by using our first model.

```r
# Train models on full training set
gss16_fit_1 <- gss16_wflow_1 %>%
  fit(gss16_train)
# Use fitted model to predict outcomes for test set
gss16_test_pred_1 <- predict(gss16_fit_1, new_data = gss16_test, type = "prob") %>%
  bind_cols(gss16_test %>% select(advfront))
# Plot ROC curve
gss16_test_pred_1 %>%
  roc_curve(
    truth = advfront,
    .pred_Agree,
    event_level = "first"
  ) %>%
  autoplot()
```



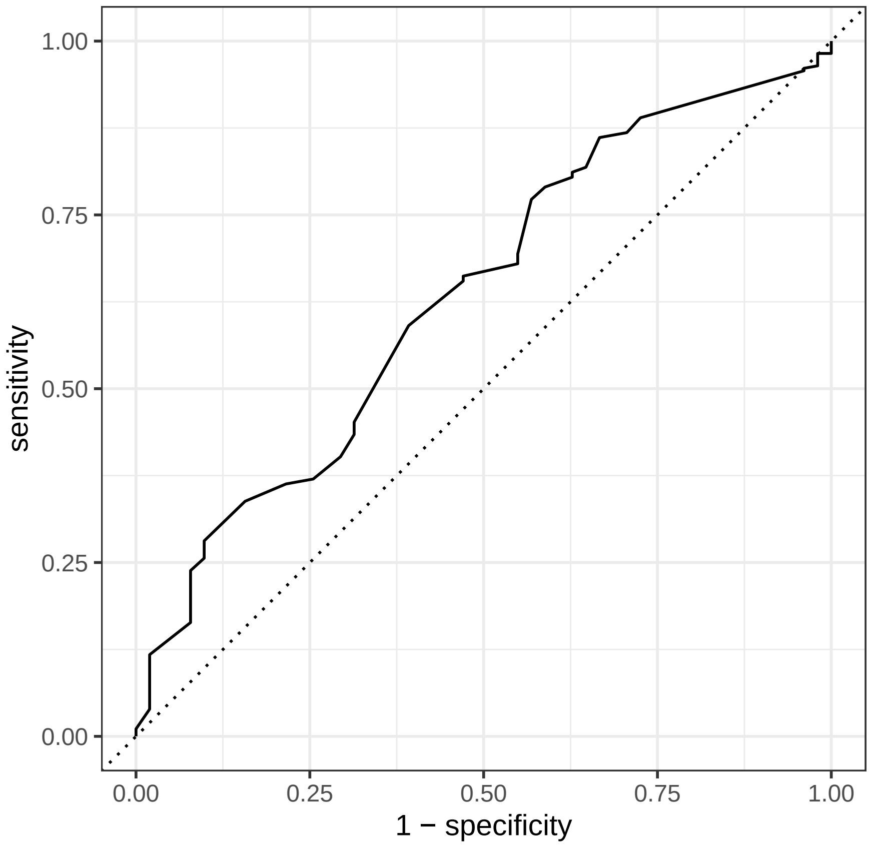
```r
# Calculate AUC
gss16_test_pred_1 %>%
  roc_auc(
    truth = advfront,
    .pred_Agree,
    event_level = "first"
  )
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.633
```

The area under the ROC curve of our first model for the testing data is 0.633.

Similarly, lets make prediction on our testing data by using our second model.

```
# Train models on full training set
gss16_fit_2 <- gss16_wflow_2 %>%
  fit(gss16_train)
# Use fitted model to predict outcomes for test set
gss16_test_pred_2 <- predict(gss16_fit_2, new_data = gss16_test, type = "prob") %>%
  bind_cols(gss16_test %>% select(advfront))
# Plot ROC curve
gss16_test_pred_2 %>%
  roc_curve(
    truth = advfront,
    .pred_Agree,
    event_level = "first"
  ) %>%
  autoplot()
```



```r
# Calculate AUC
gss16_test_pred_2 %>%
  roc_auc(
    truth = advfront,
    .pred_Agree,
    event_level = "first"
  )
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.632
```

The area under the ROC curve of our second model for the testing data is 0.632.

On our testing data, model 1 is found to have slightly higher value for the area under the ROC curve than that for the model 2. Hence we conclude that model 1 performed better than model 2 on our testing data, which is contrary to what we saw on our training data. We are seeing this difference in efficiency of our models in the training and testing data because the observations in both the set might have some differences and our models might be well in predicting better from certain set of observations than others. The fact that our first model performed well on testing data than the average for the training data is not surprising as even in some of the folds in the training data itself, our first model had higher area under the ROC curve than the average value.

Exercise 10

Now, lets explore the responses to the harassment question.

```r
gss16_har <- gss16 %>%
  filter(harass5 == "Yes" | harass5 == "No")

gss16_har %>%
  count(harass5) %>%
  mutate(harras5_prop = n / sum(n))
```

```
## # A tibble: 2 x 3
##   harass5     n harras5_prop
##   <chr>   <int>        <dbl>
## 1 No       1136        0.827
## 2 Yes       237        0.173
```

It is evident from the table above that 237 respondents agreed to have been harassed by their superiors or coworkers at job while 1136 of them disagreed to the same question. Only a small proportion of these respondents, that is 0.173, had reported being harassed in the last 5 years.

Exercise 11

Bootstrap takes random sample with replacement from the original sample, of the same size as the original sample. Then it calculate statistics of those bootstrap samples. In our case we can take numerous samples of observations for harass5 variable from our dataset and calculate the proportion of people who responded "Yes". Then we repeat the process of sampling and calculate the proportion of the people who responded "Yes" to create a bootstrap distribution. Then from the distribution that we have generated, we choose a confidence level to predict the interval for the proportion of Americans who have been harassed by their superiors or co-workers at their job in the last 5 years. In other words, by using the method of bootstrapping, we can come with a range of values for the proportion and we can be confident to a certain level that the actual proportion of Americans who have been harassed by their superiors or coworkers falls under that range.

Exercise 12

Now, lets calculate the 95% bootstrap confidence interval for our harassment question.

```r
gss16_har %>%
  # Specify the variable of interest
  specify(response = harass5, success = "Yes") %>%
  # Generate 2000 bootstrap samples
  generate(reps = 2000, type = "bootstrap") %>%
  # Calculate the proportion of each bootstrap sample
  calculate(stat = "prop") %>%
  # Find the bounds of the 95% CI
  summarize(lower = quantile(stat, 0.025),
            upper = quantile(stat, 0.975))
```

```
## # A tibble: 1 x 2
##   lower upper
##   <dbl> <dbl>
## 1 0.152 0.192
```

From the table above, we conclude that we can be 95% certain that the proportion of Americans who have been harassed by their superiors or coworkers in the last 5 years is between 0.152 and 0.192.

Exercise 13

A 90% confidence interval would be narrower than the 95% confidence interval because the lower our confidence levels are, the more precise we can be at predicting our desired statistical outcome and hence the narrower our intervals will be. We can also explain this by imagining a bootstrap distribution that is created from repeated sampling and calculating our desired statistic. To calculate the interval for the 90% confidence level, we will be selecting values of the proportion from 5% on both beginning and the end of the x-axis unlike for the 95% confidence level where we will be doing the same by only 2.5% on the beginning and the end. This will make the starting and ending value of our proportion to be closer to each other and hence of narrower interval.