Bootstrapping

Jaime Davila

3/16/2022

Introduction

Once more we will be using our `1`, `2`, and `7` dataset.

```
digits <- c("1","2","7")
train.127.tbl <- read_csv("~/Mscs 341 S22/Class/Data/train.127.csv") %>%
  mutate(y=factor(y, levels=digits))
test.127.tbl <- read_csv("~/Mscs 341 S22/Class/Data/test.127.csv") %>%
  mutate(y=factor(y, levels=digits))
```

Let's train a QDA model using our training dataset:

```
library(tidymodels)
library(discrim)
tidymodels_prefer()

qda.model <- discrim_quad() %>%
  set_engine("MASS") %>%
  set_mode("classification")

recipe <- recipe(y ~ x_1 + x_2, data=train.127.tbl)

qda.wflow <- workflow() %>%
  add_recipe(recipe) %>%
  add_model(qda.model)

qda.fit <- fit(qda.wflow, train.127.tbl)
```

1. We are interested in the proportion of `2`s that get correctly classified using our testing dataset. Calculate in an automated way this amount. *Hint* Create a confusion matrix and use the command `tidy()` to make into a tibble so that you can calculate this amount.

```
conf.mat <- augment(qda.fit, test.127.tbl) %>%
  conf_mat(truth = y, estimate = .pred_class) %>%
  tidy()

tot <- conf.mat %>%slice(4:6) %>%
    summarize(sum=sum(value)) %>% pull(sum)
correct <- conf.mat %>% slice(5) %>% pull (value)

(prop.2s <- correct/tot)
```

```
## [1] 0.699187
```

In principle the proportion of 2s that gets correctly classified might change slightly depending on the training dataset. In the remainder of this worksheet we will assess the variability of this proportion by using a collection of different training datasets.

To accomplish this we will be using the bootstrapping technique. A bootstrap dataset is obtained by sampling with replacement from the original dataset. First, we will train a model on each bootstrap dataset. The complement of our bootstrap dataset (this subset is usually called out-of-bag dataset) will be used as our testing dataset. Finally we will calculate our confusion matrix using our model and our testing dataset.

To implement this idea we will be using the function `bootstraps()` from `tidymodels`. The following exercises will guide you on how to do this.

2. Create 50 bootstraps from your training dataset using the function `bootstraps()` and store in a tibble called `bootstrap.tbl`. What is 10th bootstrap dataset? What is the 10th out-of-bag dataset? (*Hint* Use the functions `analysis()` and `assessment()`). What are the sizes of 3rd and 5th bootstrap datasets? What are the sizes of the 3rd and 5th out-of-bag datasets? Why are the sizes of the bootstrap datasets the same while the sizes of the out-of-bag datasets are different?

```
set.seed(12345)
bootstrap.tbl <- bootstraps(train.127.tbl, times=50)
analysis(bootstrap.tbl$splits[[10]])
```

```
## # A tibble: 1,601 x 3
##    y        x_1    x_2
##    <fct>  <dbl>  <dbl>
##  1 1      0      0.312
##  2 1      0      0.130
##  3 1      0.0139 0.167
##  4 2      0.0851 0.330
##  5 7      0.244  0.321
##  6 2      0.133  0.344
##  7 7      0.258  0.269
##  8 1      0      0.0426
##  9 1      0.444  0.481
## 10 2      0.139  0.287
## # ... with 1,591 more rows
```

```
assessment(bootstrap.tbl$splits[[10]])
```

```
## # A tibble: 599 x 3
##    y        x_1    x_2
##    <fct>  <dbl>  <dbl>
##  1 1      0      0.556
##  2 7      0.213  0.213
##  3 1      0.0238 0.0714
##  4 7      0.152  0.232
##  5 7      0.323  0.354
##  6 2      0.225  0.296
##  7 2      0.140  0.302
##  8 2      0      0.292
##  9 7      0.217  0.217
## 10 7      0.270  0.255
## # ... with 589 more rows
```

```
dim(analysis(bootstrap.tbl$splits[[3]]))
```

```
## [1] 1601    3
```

```
dim(analysis(bootstrap.tbl$splits[[5]]))
```

```
## [1] 1601    3
```

```
dim(assessment(bootstrap.tbl$splits[[3]]))
```

```
## [1] 608   3
```

```
dim(assessment(bootstrap.tbl$splits[[5]]))
```

```
## [1] 597   3
```

3. Define a function `calc_correct_twos_qda` that given a split will obtain testing and training datasets (remember to use `analysis()` and `assessment()`). The function will train a qda model using the testing dataset and calculate the proportion of correctly classified 2s on the testing dataset. Test your function using a couple of the bootstraps from your previous point

```
calc_correct_twos_qda <- function(split) {
 # Calculate testing/training
 train.tbl <- analysis(split)
 test.tbl <- assessment(split)

 # Create model using training dataset
 qda.model <- discrim_quad() %>%
    set_engine("MASS") %>%
    set_mode("classification")
  recipe <- recipe(y ~ x_1 + x_2, data=train.tbl)
  qda.wflow <- workflow() %>%
    add_recipe(recipe) %>%
    add_model(qda.model)
  qda.fit <- fit(qda.wflow, train.tbl)

  # Calculate % of correctly classifed twos
  conf.mat <- augment(qda.fit, test.tbl) %>%
    conf_mat(truth = y, estimate = .pred_class) %>%
    tidy()
  tot <- conf.mat %>%slice(4:6) %>%
    summarize(sum=sum(value)) %>% pull(sum)
  correct <- conf.mat %>% slice(5) %>% pull (value)
  correct/tot
}
```

```
calc_correct_twos_qda(bootstrap.tbl$splits[[5]])
```

```
## [1] 0.681592
```

```
calc_correct_twos_qda(bootstrap.tbl$splits[[10]])
```

```
## [1] 0.7204301
```

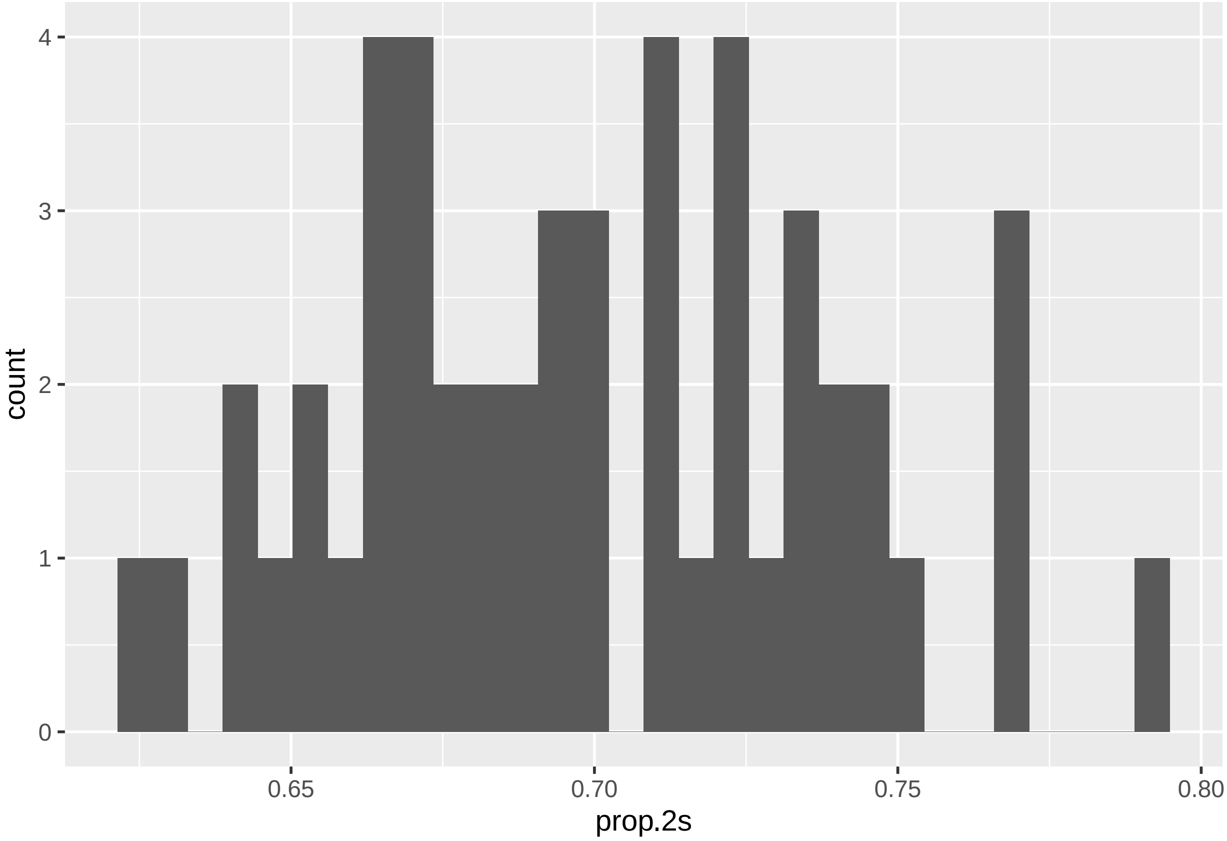
Finally we need to apply the function `calc_correct_twos_qda()` on all the splits from `bootstrap.tbl`. Notice that the type of the column `splits` is a `list`, so we can use the function `map_dbl()`. `map_dbl(lst, f)` applies the function `f()` to all the elements of `lst` and outputs a `double` (that is why the suffix `_dbl`). We can do this as follows:

```
boots.values.tbl <- bootstrap.tbl %>%
  mutate(prop.2s = map_dbl(splits, calc_correct_twos_qda))
```

4. Plot the histogram of the proportion of correctly classified 2s. Calculate the mean and standard

deviation of this metric. Does the histogram, mean and standard deviation change a lot if you use a different set of 200 boostraps?

```
ggplot(boots.values.tbl, aes(prop.2s)) +
  geom_histogram()
```



```
mean(boots.values.tbl$prop.2s)
```

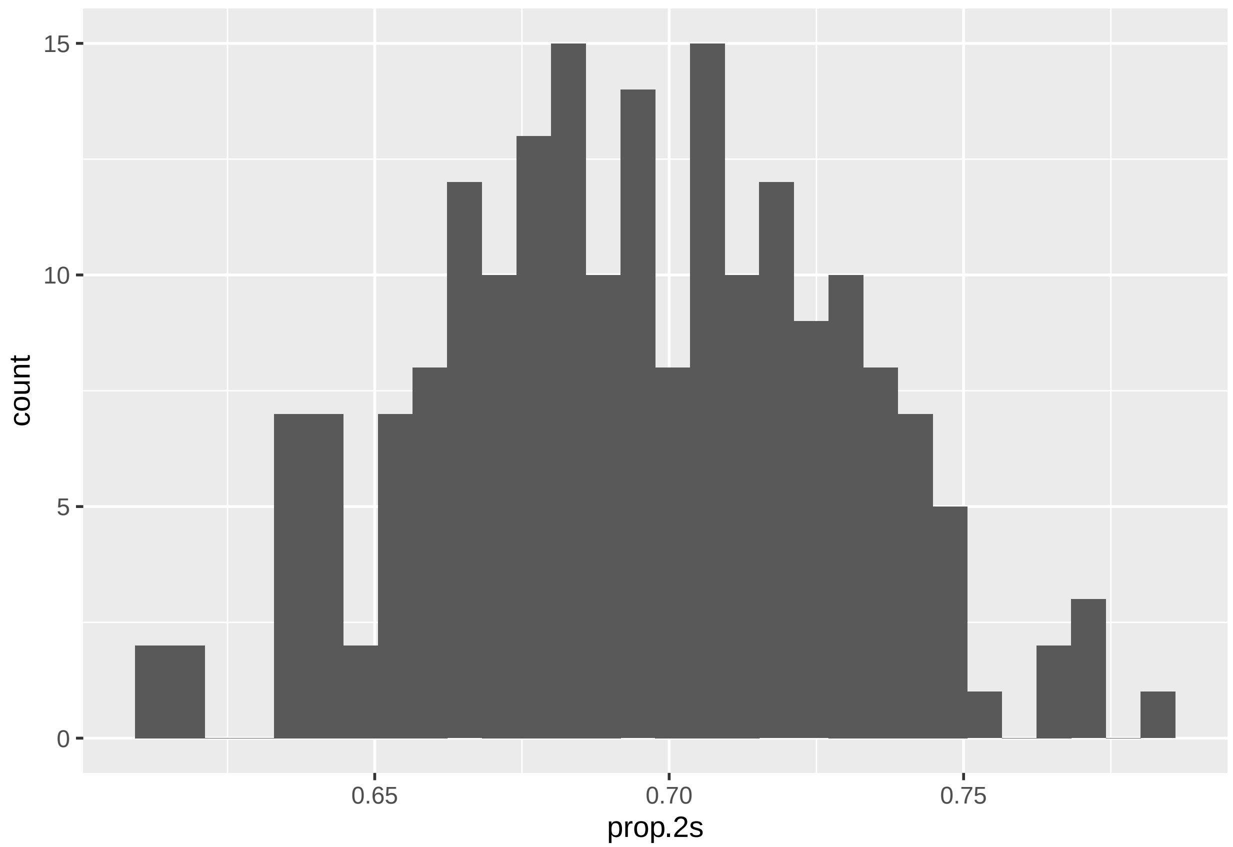
```
## [1] 0.6995363
```

```
sd(boots.values.tbl$prop.2s)
```

```
## [1] 0.0393577
```

```
bootstrap.tbl <- bootstraps(train.127.tbl, times=200)
boots.values.tbl <- bootstrap.tbl %>%
  mutate(prop.2s = map_dbl(splits, calc_correct_twos_qda))

ggplot(boots.values.tbl, aes(prop.2s)) +
  geom_histogram()
```



```
mean(boots.values.tbl$prop.2s)
```

```
## [1] 0.6942116
```

```
sd(boots.values.tbl$prop.2s)
```

```
## [1] 0.03418526
```

We will be using more bootstrapping after the break when we introduce more advance modeling approaches, so stay tuned!

Remember:

No HW for next week (3/21-3/25)

Exam 1: In-class (3/24). Covers Ch. 2,3,4 and 5.1 of ISLR plus R-bootcamp (week 1 of class) and tidymodels!

Sample in-class exam on Tu 3/22