Crosss Validation of Class Predictions

Kevin R. Coombes

July 20, 2022

Contents

1 Introduction

When building models to make predictions of a binary outcome from omics-scale data, it is especially useful to thoroughly cross-validate those models by repeatedly splitting the data into training and test sets. The *CrossValidate* package provides tools to simplify this procedure.

2 A Simple Example

We start by loading the package

```
> library(CrossValidate)
```

Now we simulate a data set with no structure that we can use to test the methods.

```
> set.seed(123456)
> nFeatures <- 1000
> nSamples <- 60
> pseudoclass <- factor(rep(c("A", "B"), each = 30))
> dataset <- matrix(rnorm(nFeatures * nSamples), nrow = nFeatures)
```

Now we pick a model that we would like to cross-validate. To start, we will use K nearest neighbors (KNN) with $K = 3$.

```
> model <- modeler5NN
```

The we invoke the cross-validation procedure.

```
> cv <- CrossValidate(model, dataset, pseudoclass, frac = 0.6, nLoop = 30)
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
[1] 7
[1] 8
[1] 9
[1] 10
[1] 11
[1] 12
[1] 13
[1] 14
[1] 15
[1] 16
[1] 17
[1] 18
[1] 19
[1] 20
[1] 21
[1] 22
[1] 23
[1] 24
[1] 25
[1] 26
[1] 27
[1] 28
[1] 29
[1] 30
```

By default (`verbose = TRUE`), the cross validation procedure prints out a counter for each iteration. This behavior can be overridden by setting `verbose = FALSE`.

```
> summary(cv)

---------------
Cross-validation was performed using 60 percent of the data for
training. The data set was randomly split into training and testing
sets 30 times.

Training Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.6111111 0.3888889 0.5277778 0.5217391 0.5384615
1st Qu. 0.7222222 0.5694444 0.6944444 0.6570513 0.7222222
Median  0.7777778 0.6388889 0.7222222 0.6842105 0.7333333
Mean    0.7777778 0.6444444 0.7111111 0.6938578 0.7500045
3rd Qu. 0.8333333 0.7222222 0.7222222 0.7222222 0.7857143
Max.    0.9444444 0.8888889 0.8333333 0.8571429 0.9000000
```

```
Validation Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.4166667 0.1666667 0.2916667 0.3333333 0.2222222
1st Qu. 0.4375000 0.3333333 0.4166667 0.4285714 0.3812500
Median  0.5416667 0.3333333 0.4583333 0.4580420 0.4615385
Mean    0.5555556 0.3972222 0.4763889 0.4815291 0.4659953
3rd Qu. 0.6666667 0.5000000 0.5312500 0.5250000 0.5340909
Max.    0.7500000 0.6666667 0.6666667 0.6666667 0.6666667
```

The summary reports the performance separately on the training data and the testing data. In this case, KNN overfits the training data (getting roughly 70% of the "predictions" correct) but is no better than coin toss on the test data.

3 Testing Multiple Models

A primary advantage of defining a common interface to different classification methods is that you can write code that tests them all in exactly the same way. For example, let's suppose that we want to compare the KNN method above to the method of compound covariate predictors. We can then do the following.

```
> models <- list(KNN = modeler5NN, CCP = modelerCCP)
> results <- lapply(models, CrossValidate,
+                   data = dataset, status = pseudoclass,
+                   frac = 0.6, nLoop = 30, verbose = FALSE)
> lapply(results, summary)
```

```
$KNN
---------------
Cross-validation was performed using 60 percent of the data for
training. The data set was randomly split into training and testing
sets 30 times.

Training Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.4444444 0.3333333 0.5833333 0.5555556 0.5652174
1st Qu. 0.7222222 0.5000000 0.6666667 0.6189459 0.6764706
Median  0.7777778 0.6666667 0.6944444 0.6830144 0.7573529
Mean    0.7851852 0.6314815 0.7083333 0.6899356 0.7575777
3rd Qu. 0.8750000 0.7222222 0.7708333 0.7500000 0.8125000
Max.    1.0000000 0.9444444 0.8611111 0.9333333 1.0000000

Validation Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.2500000 0.0000000 0.2916667 0.3076923 0.0000000
1st Qu. 0.5000000 0.2500000 0.4270833 0.4392361 0.3812500
Median  0.6666667 0.3750000 0.5416667 0.5294118 0.5555556
Mean    0.6111111 0.3805556 0.4958333 0.4994913 0.4786667
3rd Qu. 0.7291667 0.5000000 0.5833333 0.5607639 0.5714286
```

```
Max.    0.8333333 0.7500000 0.5833333 0.6250000 0.6666667

$CCP
---------------
Cross-validation was performed using 60 percent of the data for
training. The data set was randomly split into training and testing
sets 30 times.

Training Accuracy:
        sens spec acc ppv npv
Min.       1    1   1   1   1
1st Qu.    1    1   1   1   1
Median     1    1   1   1   1
Mean       1    1   1   1   1
3rd Qu.    1    1   1   1   1
Max.       1    1   1   1   1

Validation Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.1666667 0.2500000 0.2500000 0.2000000 0.2500000
1st Qu. 0.2708333 0.3333333 0.3333333 0.3571429 0.3392857
Median  0.3750000 0.4166667 0.4166667 0.4083333 0.4285714
Mean    0.3833333 0.4388889 0.4111111 0.4034968 0.4105820
3rd Qu. 0.4791667 0.5625000 0.4583333 0.4597902 0.4666667
Max.    0.6666667 0.6666667 0.5833333 0.6000000 0.6000000
```

The performance of KNN with this set of training-test splits is simila to the previous set. The CCP metod, by contrast, behaves much worse. It perfectly fits (and so overfits) the training data and consequently actually manages to do *worse* than chance on the test data.

4 Filtering and Pruning

Having a common interface also lets us write code that combines the same modeling method with different algorothms to filter genes (by something like mean expression, for example) or to perform feature selection (using univariate t-tests, for example). Many such methods are prvoided by the *Modeler* package on which *CrossValidate* depends. Here we show how to combine the KNN method with several different methods to preprocess the set of features.

Here we show how to do this the wrong way.

```
> pruners <- list(ttest = fsTtest(fdr = 0.05, ming = 100),
+                 cor = fsPearson(q = 0.90),
+                 ent = fsEntropy(q = 0.90, kind = "information.gain"))
> for (p in pruners) {
+   pdata <- dataset[p(dataset, pseudoclass),]
+   cv <- CrossValidate(model, pdata, pseudoclass, 0.6, 30, verbose=FALSE)
+   show(summary(cv))
+ }
```

```
---------------
Cross-validation was performed using 60 percent of the data for
training. The data set was randomly split into training and testing
sets 30 times.

Training Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.9444444 0.7777778 0.8888889 0.8181818 0.9411765
1st Qu. 0.9444444 0.8888889 0.9444444 0.9000000 0.9444444
Median  0.9722222 0.9444444 0.9444444 0.9444444 0.9736842
Mean    0.9722222 0.9296296 0.9509259 0.9355406 0.9722853
3rd Qu. 1.0000000 0.9444444 0.9722222 0.9473684 1.0000000
Max.    1.0000000 1.0000000 1.0000000 1.0000000 1.0000000

Validation Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.7500000 0.6666667 0.7916667 0.7500000 0.7692308
1st Qu. 0.8541667 0.8333333 0.8750000 0.8333333 0.8678571
Median  0.9166667 0.8750000 0.9166667 0.8869048 0.9166667
Mean    0.9277778 0.8750000 0.9013889 0.8875963 0.9286902
3rd Qu. 1.0000000 0.9166667 0.9479167 0.9230769 1.0000000
Max.    1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
---------------
Cross-validation was performed using 60 percent of the data for
training. The data set was randomly split into training and testing
sets 30 times.

Training Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.8888889 0.8333333 0.8888889 0.8571429 0.8888889
1st Qu. 0.9444444 0.8888889 0.9444444 0.9000000 0.9451754
Median  1.0000000 0.9444444 0.9444444 0.9459064 1.0000000
Mean    0.9722222 0.9425926 0.9574074 0.9464547 0.9731774
3rd Qu. 1.0000000 1.0000000 0.9722222 1.0000000 1.0000000
Max.    1.0000000 1.0000000 1.0000000 1.0000000 1.0000000

Validation Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.8333333 0.5833333 0.7916667 0.7058824 0.8461538
1st Qu. 0.9166667 0.7500000 0.8437500 0.7892857 0.8916667
Median  0.9166667 0.8333333 0.8750000 0.8571429 0.9090909
Mean    0.9305556 0.8361111 0.8833333 0.8594585 0.9298688
3rd Qu. 1.0000000 0.9166667 0.9166667 0.9090909 1.0000000
Max.    1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
---------------
Cross-validation was performed using 60 percent of the data for
training. The data set was randomly split into training and testing
sets 30 times.
```

```
Training Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.7222222 0.5555556 0.7500000 0.6800000 0.7619048
1st Qu. 0.8333333 0.7777778 0.8333333 0.7921053 0.8333333
Median  0.9444444 0.8333333 0.8611111 0.8416667 0.9333333
Mean    0.9092593 0.8037037 0.8564815 0.8271060 0.9067213
3rd Qu. 0.9444444 0.8750000 0.8888889 0.8642857 0.9411765
Max.    1.0000000 0.8888889 0.9444444 0.9000000 1.0000000

Validation Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.5000000 0.3333333 0.5833333 0.5555556 0.5833333
1st Qu. 0.5833333 0.5833333 0.6250000 0.6266447 0.6206294
Median  0.7500000 0.6666667 0.6666667 0.6666667 0.7238095
Mean    0.7416667 0.6500000 0.6958333 0.6859249 0.7373619
3rd Qu. 0.8333333 0.7500000 0.7500000 0.7318182 0.8333333
Max.    1.0000000 0.9166667 0.8750000 0.8888889 1.0000000
```

We can tell that this method is wrong because the validation accuracy is much better than chance—which is impossible on a dataset without any true structure. The problem is that we have aplied the fature selection method to the combined (training plus test) dataset, which allows information from the test data to creep into the model building step.

Now we can do it the right way, with the feature selection step included inside the cross-validation loop.

```
> for (p in pruners) {
+   cv <- CrossValidate(model, dataset, pseudoclass, 0.6, 30,
+                       prune=p, verbose=FALSE)
+   show(summary(cv))
+ }

---------------
Cross-validation was performed using 60 percent of the data for
training. The data set was randomly split into training and testing
sets 30 times.

Training Accuracy:
             sens      spec       acc      ppv       npv
Min.    0.9444444 0.8888889 0.9444444 0.900000 0.9444444
1st Qu. 1.0000000 1.0000000 1.0000000 1.000000 1.0000000
Median  1.0000000 1.0000000 1.0000000 1.000000 1.0000000
Mean    0.9944444 0.9907407 0.9925926 0.991306 0.9946394
3rd Qu. 1.0000000 1.0000000 1.0000000 1.000000 1.0000000
Max.    1.0000000 1.0000000 1.0000000 1.000000 1.0000000

Validation Accuracy:
              sens      spec       acc       ppv       npv
Min.    0.08333333 0.1666667 0.2083333 0.1111111 0.2500000
1st Qu. 0.41666667 0.3333333 0.3750000 0.3914027 0.3774038
```

```
Median  0.50000000 0.4166667 0.4583333 0.4686275 0.4641026
Mean    0.49722222 0.4222222 0.4597222 0.4545720 0.4586675
3rd Qu. 0.58333333 0.5000000 0.5312500 0.5197368 0.5288462
Max.    0.83333333 0.5833333 0.6250000 0.6153846 0.6363636
---------------
Cross-validation was performed using 60 percent of the data for
training. The data set was randomly split into training and testing
sets 30 times.

Training Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.9444444 0.9444444 0.9722222 0.9473684 0.9473684
1st Qu. 1.0000000 1.0000000 0.9791667 1.0000000 1.0000000
Median  1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
Mean    0.9962963 0.9888889 0.9925926 0.9894737 0.9964912
3rd Qu. 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
Max.    1.0000000 1.0000000 1.0000000 1.0000000 1.0000000

Validation Accuracy:
             sens       spec       acc       ppv       npv
Min.    0.1666667 0.08333333 0.2500000 0.2500000 0.1250000
1st Qu. 0.4166667 0.33333333 0.4166667 0.4285714 0.4041667
Median  0.5000000 0.37500000 0.4791667 0.4852941 0.4833333
Mean    0.5444444 0.41388889 0.4791667 0.4810754 0.4798958
3rd Qu. 0.6666667 0.50000000 0.5416667 0.5323529 0.5714286
Max.    0.9166667 0.83333333 0.6666667 0.7000000 0.6666667
---------------
Cross-validation was performed using 60 percent of the data for
training. The data set was randomly split into training and testing
sets 30 times.

Training Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.6666667 0.7222222 0.8055556 0.7619048 0.7500000
1st Qu. 0.8333333 0.8888889 0.8888889 0.9000000 0.8517857
Median  0.8888889 0.9444444 0.9166667 0.9444444 0.9000000
Mean    0.8962963 0.9314815 0.9138889 0.9355316 0.9067610
3rd Qu. 0.9444444 1.0000000 0.9444444 1.0000000 0.9473684
Max.    1.0000000 1.0000000 1.0000000 1.0000000 1.0000000

Validation Accuracy:
             sens      spec       acc       ppv       npv
Min.    0.1666667 0.1666667 0.3750000 0.3846154 0.2857143
1st Qu. 0.4166667 0.5000000 0.5000000 0.5000000 0.5000000
Median  0.5000000 0.5416667 0.5416667 0.5505051 0.5419580
Mean    0.5361111 0.5638889 0.5500000 0.5662765 0.5542882
3rd Qu. 0.6666667 0.6666667 0.5833333 0.6187500 0.6000000
Max.    0.9166667 1.0000000 0.7500000 1.0000000 0.8571429
```