Evaluation Metrics for Classification Models

Brooke Anderson

January 26, 2016

```
knitr::opts_knit$set(fig.path = '../figures/EvalMetrics-',
                     root.dir = '..')
```

Note: I was tired of my Rmd documents, which I put in the sub-directory "Rscripts", running from a different working directory than my R session when I'm working on this project (which was running from "Titanic", the parent directory of that "Rscripts directory"). This was making me have to use one relative pathname when I knitted and a different when I tested code chunks interactively. Therefore, I used the `root.dir` option in `opts_knit` to change the working directory that will be used when I knit to the parent directory of where the Rmd file is saved.

Load required libraries.

```
library(dplyr)  ## Data wrangling
library(klaR) ## Includes `NaiveBayes` function
library(caret)
library(pROC)
```

Bring in the data:

```
train <- read.csv("data/train.csv") %>%
  mutate(Survived = factor(Survived),
         Pclass = factor(Pclass),
         Name = as.character(Name),
         Sex = factor(Sex))
test <- read.csv("data/test.csv") %>%
  mutate(Pclass = factor(Pclass),
         Name = as.character(Name),
         Sex = factor(Sex))
```

For this, I'll do different evaluation metrics for a Naive Bayes with only sex as a predictor. I think that the first of these might be the benchmark models for Kaggle, with the "predict survival if woman, death otherwise" model that some folks in class put in.

This time, I'll try using the `NaiveBayes` function from the `klaR` package. Based on Kuhn and Johnson, in comparing this with the `e1071` package:

> "Both offer Laplace corrections, but the version in the `klaR` package has the option of using conditional density estimates that are more flexible."

```
nb_sex <- NaiveBayes(Survived ~ Sex, data = train)
```

This function's output provides some of the same values as the other Naive Bayes function I tried, including $P(Y)$ and $P(X|Y)$:

```r
nb_sex$apriori
```

```
## grouping
##         0         1
## 0.6161616 0.3838384
```

```r
nb_sex$tables
```

```
## $Sex
##         var
## grouping    female      male
##        0 0.1475410 0.8524590
##        1 0.6812865 0.3187135
```

Evidently, you can use `predict` with this function, including with an optional argument specifying `newdata`. To predict with the training data:

```r
pred_train <- predict(nb_sex, newdata = train)
```

This prediction includes two elements: the class predictions (0 = died; 1 = survived) and the posterior class probabilities:

```r
names(pred_train)
```

```
## [1] "class"     "posterior"
```

```r
head(pred_train$class)
```

```
## [1] 0 1 1 1 0 0
## Levels: 0 1
```

```r
head(pred_train$posterior)
```

```
##             0         1
## [1,] 0.8110919 0.1889081
## [2,] 0.2579618 0.7420382
## [3,] 0.2579618 0.7420382
## [4,] 0.2579618 0.7420382
## [5,] 0.8110919 0.1889081
## [6,] 0.8110919 0.1889081
```

I checked to see if this always predicts that women survive and men die:

```r
table(train$Sex, pred_train$class)
```

```
##
##            0   1
##   female   0 314
##   male   577   0
```

Yep. So, this should give the results from the benchmark "Sex" model in the Kaggle competition.

To create predictions to submit to Kaggle:

```
pred_test <- predict(nb_sex, newdata = test)
```

And write them out to a comma-separated file. Note that the `pred_test$class` is saved as a factor variable, so if you don't convert to a character and then a number using the `as.*` functions, you might get outputs of 1s and 2s instead of 0s and 1s, which would cause problems when you submit to Kaggle.

```
out <- cbind(test$PassengerId,
             as.numeric(as.character(pred_test$class)))
colnames(out) <- c("PassengerId","Survived")
write.csv(out, file = "predictions/nb_sex.csv", row.names = FALSE)
```

Evaluation metrics

Accuracy

Kaggle is judging this competition on accuracy, so that's the main metrics we've been using. In the training dataset, the accuracy of this model is:

```
mean(train$Survived == pred_train$class)
```

```
## [1] 0.7867565
```

To get the accuracy on the test data set, I have to submit the file I wrote out, "nb_sex.csv", to Kaggle. I did, and the accuracy based on the test data for the Public Leaderboard was 0.76555. This is indeed the score for the Gender-Based Model benchmark and ties my best score to date.

Sensitivity / specificity

You can get a confusion matrix to use to calculate many of these metrics pretty simply by taking a table of the model predictions versus the observed labels:

```
table(pred_train$class, train$Survived)
```

```
##
##       0   1
##   0 468 109
##   1  81 233
```

You can also use functions from R packages to calculate everything. The `caret` package is a bit of a Swiss Army knife for machine learning. You can use it, among other things, to check sensitivity and specificity, and a variety of other model evaluation metrics using its `confusionMatrix` function:

```
nb_eval <- confusionMatrix(data = pred_train$class,
                           reference = train$Survived,
                           positive = "1")
```

This function outputs an object that includes:

A confusion matrix:

```
nb_eval$table
```

```
##           Reference
## Prediction   0   1
##          0 468 109
##          1  81 233
```

Overall accuracy:

```
nb_eval$overall["Accuracy"]
```

```
##  Accuracy
## 0.7867565
```

Sensitivity and specificity:

```
nb_eval$byClass["Sensitivity"]
```

```
## Sensitivity
##   0.6812865
```

```
nb_eval$byClass["Specificity"]
```

```
## Specificity
##    0.852459
```

Positive and negative predictive values

The output of the `confusionMatrix` function also provides positive predictive value (PPV) and negative predictive value (NPV):

```
nb_eval$byClass["Pos Pred Value"]
```

```
## Pos Pred Value
##      0.7420382
```

```
nb_eval$byClass["Neg Pred Value"]
```

```
## Neg Pred Value
##      0.8110919
```

Kappa

The output of the `confusionMatrix` function also provides a Kappa value:

```
nb_eval$overall["Kappa"]
```
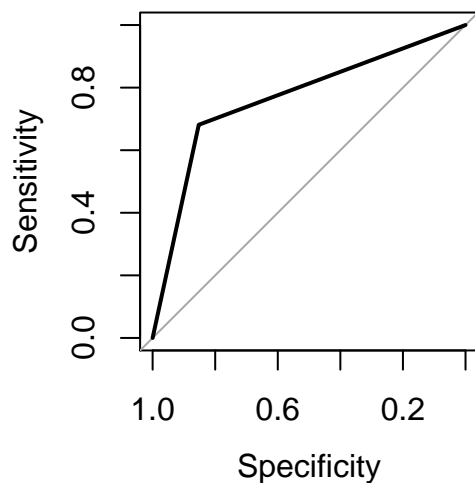
```
##     Kappa
## 0.542113
```

ROC Curve

You can use the `roc` function in the `pROC` package to create Receiver Operating Characteristic (ROC) curves:

```
roc_nb_sex <- roc(response = as.numeric(as.character(train$Survived)),
                  predictor = as.numeric(as.character(pred_train$class)))
```

This can be plotted:

```
plot(roc_nb_sex)
```


(As a reminder, to get the help file for something like this that is a method for a class, use, e.g., `?plot.roc`.)

This can be used to calculate the Area Under the Curve (AUC):

```
auc(roc_nb_sex)
```

```
## Area under the curve: 0.7669
```

To think about

In the `confusionMatrix` output, what are:

- The No Informaiton Rate
- Prevalence
- Detection Rate
- Detection Prevalence
- Balanced Accuracy

Other things to think about:

- What are some different evaluation metrics used by other Kaggle classification competitions?
- Think of some examples of predictive tasks where Accuracy would be a poor metric to use to evaluate models.
- How do these metrics work when you are classifying into > 2 classes?