Logistic Regression Performance

Fundamental Techniques in Data Science

Utrecht University

Kyle M. Lang

Department of Methodology & Statistics
Utrecht University

Outline

Confusion Matrix

ROC Curve

Alternative Performance Measures

Example Model

First, we'll refit the logistic regression model that predicts the chances of Titanic passengers surviving based on their age, sex, and ticket price

```
## Read the data:
titanic <- readRDS(here::here("data", "titanic.rds"))

## Estimate the logistic regression model:
glmFit <- glm(survived ~ age + sex + fare,
              data = titanic,
              family = "binomial")

## Save the linear predictor estimates:
titanic$etaHat <- predict(glmFit, type = "link")
```

Example Model

```
partSummary(glmFit, -1)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.837621   0.215121   3.894 9.87e-05
age         -0.007404   0.006040  -1.226     0.22
sexmale     -2.392422   0.171288 -13.967  < 2e-16
fare         0.011586   0.002338   4.955 7.23e-07

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.8  on 886  degrees of freedom
Residual deviance:  881.4  on 883  degrees of freedom
AIC: 889.4

Number of Fisher Scoring iterations: 5
```

Confusion Matrix

One of the most direct ways to evaluate classification performance is the *confusion matrix*.

```
library(magrittr)

## Add predictions to the dataset:
titanic %<>%
  mutate(piHat = predict(glmFit, type = "response"),
         yHat = as.factor(ifelse(piHat <= 0.5, "no", "yes"))
        )
```

	True	
Predicted	no	yes
no	458	106
yes	87	236

Confusion Matrix of Predicted Survival

Confusion Matrix

Each cell in the confusion matrix represents a certain classification result.

	True	
Predicted	Died	Survived
Died	True Negative	False Negative
Survived	False Positive	True Positive

Confusion Matrix of Predicted Survival

- **TP**: Correctly predict survival
- **TN**: Correctly predict death
- **FP**: Predict survival for dead people
- **FN**: Predict death for survivors

Confusion Matrix

```
library(caret)

cMat <- titanic %$% confusionMatrix(data = yHat, reference = survived)

cMat$table

          Reference
Prediction  no yes
       no  458 106
       yes  87 236

cMat$overall

      Accuracy          Kappa  AccuracyLower  AccuracyUpper
  7.824126e-01   5.359709e-01   7.537802e-01   8.091549e-01
  AccuracyNull AccuracyPValue  McnemarPValue
  6.144307e-01   7.471405e-27   1.950898e-01
```

Confusion Matrix

```
cMat$byClass

         Sensitivity          Specificity
           0.8403670            0.6900585
      Pos Pred Value       Neg Pred Value
           0.8120567            0.7306502
           Precision                Recall
           0.8120567            0.8403670
                  F1           Prevalence
           0.8259693            0.6144307
      Detection Rate Detection Prevalence
           0.5163472            0.6358512
   Balanced Accuracy
           0.7652127
```

Summaries of the Confusion Matrix

Accuracy = (TP + TN) / (P + N)

- In our example, Accuracy = 0.78
- 78% are correctly classified

Error Rate = (FP + FN) / (P + N) = 1 - Accuracy

- In our example, Error Rate = 0.22
- 22% are incorrectly classified

Sensitivity = TP / (TP + FN)

- In our example, Sensitivity = 0.84
- 84% of survivors are correctly classified

Specificity = TN / (TN + FP)

- In our example, Specificity = 0.69
- 69% of deaths are correctly classified

Summaries of the Confusion Matrix

False Positive Rate (FPR) = FP / (TN + FP) = 1 - Specificity

- In our example, FPR = 0.31
- 31% of deaths are incorrectly classified as survivors

Positive Predictive Value (PPV) = TP / (TP + FP)

- In our example, PPV = 0.81
- There is an 81% chance that a passenger classified as a survivor was classified correctly

Negative Predictive Value (NPV) = TN / (TN + FN)

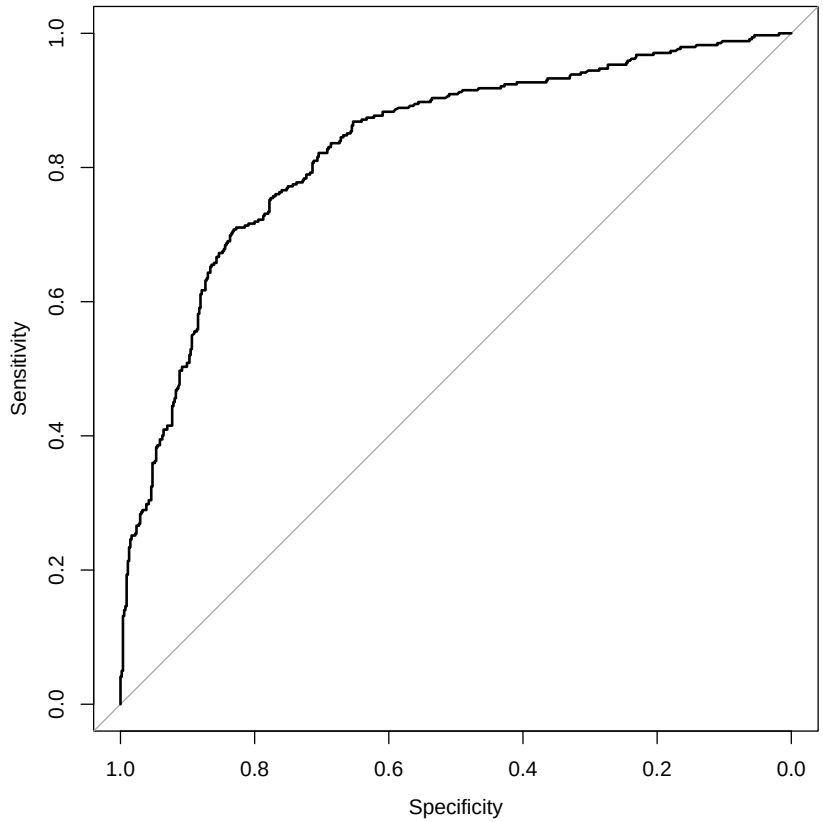
- In our example, NPV = 0.73
- There is a 73% chance that a passenger classified as dying was classified correctly

ROC Curve

```
rocData <- titanic %$%
  pROC::roc(survived, piHat)
plot(rocData)
```

A *receiver operating characteristic* (ROC) curve illustrates the diagnostic ability of a binary classifier for all possible values of the classification threshold.

- The ROC curve plots sensitivity against specificity at different threshold values.



ROC Curve

The *area under the ROC curve* (AUC) is a one-number summary of the potential performance of the classifier.

- The AUC does not depend on the classification threshold.

```
pROC::auc(rocData)

Area under the curve: 0.8298
```

According to Mandrekar (2010):

- AUC value from 0.7 – 0.8: Acceptable
- AUC value from 0.8 – 0.9: Excellent
- AUC value over 0.9: Outstanding

Threshold Selection

We can use numerical methods to estimate an optimal threshold value.

```
library(OptimalCutpoints)

ocOut <- optimal.cutpoints(X = "piHat",
                           status = "survived",
                           tag.healthy = "no",
                           data = titanic,
                           method = "ROC01"
                           )
```

Threshold Selection

```
partSummary(ocOut, -1)

Area under the ROC curve (AUC):  0.83 (0.802, 0.858)

CRITERION: ROC01
Number of optimal cutoffs: 1

                     Estimate
cutoff              0.2360978
Se                  0.7543860
Sp                  0.7761468
PPV                 0.6789474
NPV                 0.8343195
DLR.Positive        3.3700029
DLR.Negative        0.3164531
FP                122.0000000
FN                 84.0000000
Optimal criterion   0.1104365
```

Optimized Confusion Matrix

```
## Extract the optimal cutpoint:
optThresh <- ocOut$ROC01$Global$optimal.cutoff$cutoff

## Add optimized predictions to the dataset:
titanic %<>%
  mutate(yHat2 = as.factor(ifelse(piHat <= optThresh, "no", "yes")))

## Compute the optimized confusion matrix:
cMat2 <- titanic %$% confusionMatrix(data = yHat2, reference = survived)

cMat2$table

          Reference
Prediction  no yes
       no  423  85
       yes 122 257
```

Compare Thresholds

	True	
Predicted	Died	Survived
Died	458	106
Survived	87	236

Naive Threshold

	True	
Predicted	Died	Survived
Died	423	85
Survived	122	257

Optimized Threshold

	Acc	Sen	Spe	PPV	NPV
Naive	0.782	0.840	0.690	0.812	0.731
Optimized	0.767	0.776	0.751	0.833	0.678

Alternative Performance Measures

Measuring classification performance from a confusion matrix can be problematic.

- Sometimes too coarse.

We can also base our error measure on the residual deviance with the *Cross-Entropy Error*:

$$CEE = -N^{-1} \sum_{n=1}^{N} Y_n \ln(\hat{\pi}_n) + (1 - Y_n) \ln(1 - \hat{\pi}_n)$$

- The CEE is sensitive to classification confidence.
- Stronger predictions are more heavily weighted.
- The CEE doesn't depend on the classification threshold

Benefits of CEE

The misclassification rate is a naïvely appealing option.

- The proportion of cases assigned to the wrong group

Consider two perfect classifiers:

1. $P(\hat{Y}_n = 1|Y_n = 1) = 0.90, P(\hat{Y}_n = 1|Y_n = 0) = 0.10, n = 1, 2, \ldots, N$
2. $P(\hat{Y}_n = 1|Y_n = 1) = 0.55, P(\hat{Y}_n = 1|Y_n = 0) = 0.45, n = 1, 2, \ldots, N$

Both of these classifiers will have the same misclassification rate.

- Neither model ever makes an incorrect group assignment.

The first model will have a lower CEE.

- The classifications are made with higher confidence.
- $CEE_1 = 0.105, CEE_2 = 0.598$

CEE Example

Fit an alternative model by adding ticket class to our running example.

```
glmFit2 <- update(glmFit, ". ~ . + class")
partSummary(glmFit2, 2:4)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.5933885  0.4261192   8.433  < 2e-16
age         -0.0340848  0.0072237  -4.718 2.38e-06
sexmale     -2.5851093  0.1878983 -13.758  < 2e-16
fare         0.0004142  0.0021066   0.197    0.844
class2nd    -1.1739314  0.2911585  -4.032 5.53e-05
class3rd    -2.4261391  0.2936692  -8.261  < 2e-16

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.77  on 886  degrees of freedom
Residual deviance:  801.56  on 881  degrees of freedom
AIC: 813.56
```

CEE Example

Use the CEE to compare models.

```
library(MLmetrics)

# Create a numeric version of our outcome variable:
survived <- ifelse(titanic$survived == "yes", 1, 0)

# Compute the CEE for the smaller model:
LogLoss(y_pred = predict(glmFit, type = "response"), y_true = survived)
```

```
[1] 0.496846
```

```
# Compute the CEE for the larger model:
LogLoss(y_pred = predict(glmFit2, type = "response"), y_true = survived)
```

```
[1] 0.451835
```

Don't get too excited.

- We're using fitted values, not out-of-sample predictions.
- The larger model will always have a lower training-set CEE.

References

Mandrekar, J. N. (2010). Receiver operating characteristic curve in diagnostic test assessment. *Journal of Thoracic Oncology*, *5*(9), 1315–1316.