Logistic Regression - Measuring Model Performance

Ryan Miller

Introduction

- Logistic regression is a generalized linear model used when the outcome variable is binary
  - In comparison to other binary classification approaches (ie: machine learning), logistic regression models are *intrepretable* and allow for *statistical inference*
- Last week, our focus was understanding the roles of individual predictor variables within these models
  - This week, our focus will be on holistically evaluating and comparing different models

Credit Card Payments

- For risk management purposes, credit card companies will predict the likelihood of their customers missing a payment in a given month
  - We'll use data from publicly available research involving a cross-sectional sample of 30,000 customers from a major credit company in Taiwan

```
fr <- read.csv("https://remiller1450.github.io/data/Credit.csv")
table(fr$MISSED_PAYMENT)
```

```
##
##     0     1
## 23364  6636
```

Summarizing a Logistic Regression Model

- In this application, the roles played by individual predictors are likely secondary to the overall performance of the model
  - If the company cannot model missed payments with a reasonable degree of reliability, any inferences on the explanatory variables are irrelevant
- How might you measure/quantify how well a model is able to predict missed payments?

Classification Accuracy

- A simple measure of a model's ability is *classification accuracy*
  - This is found by mapping the model's predicted probabilities for each data-point to predicted binary outcomes, then tallying the proportion of these predictions that match the observed categorical outcome

Classification Accuracy

- The code below fits the logistic regression model `MISSED_PAYMENT ~ BILL_AMT1` and maps predicted probabilities exceeding 0.5 to a predicted missed payment
  - The model achieves approximately 78% accuracy, so is it a good model?

```
m <- glm(MISSED_PAYMENT ~ BILL_AMT1, data = fr, family = "binomial")
pred_probs <- predict(m, type = "response")
t = 0.5
pred_class <- ifelse(pred_probs > t, 1, 0)
sum(pred_class == fr$MISSED_PAYMENT)/nrow(fr)
```

```
## [1] 0.7788
```

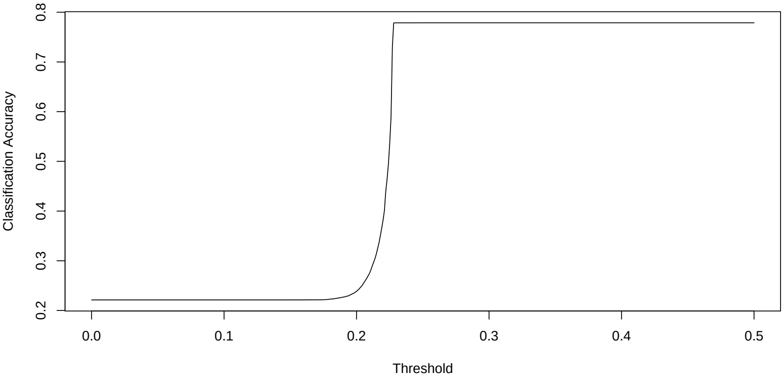
Problems with Accuracy

There are a few issues we should consider when evaluating whether a model's classification accuracy is truly indicative of strong predictive ability

1) What accuracy could be achieved by simply predicting every data-point belongs to the most common category?
2) Is the model overfit to the sample data? That is, will the model's *out-of-sample* accuracy be substantially lower than its *in-sample* accuracy

Manipulating the Decision Threshold

- A simple starting point, is to change the threshold for a missed payment from $t = 0.5$ to something lower
  - Unfortunately this doesn't do much for these data, adjusting the threshold downward can only lead to lower overall accuracy as more non-missed payments are classified as missed



Cohen's Kappa

- Cohen's Kappa is an alternative that normalizes for class imbalance in the data

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

- p_o is the *observed accuracy* of the model being evaluated
- p_e is the *expected accuracy* that could be achieved by predicting every data-point belongs to the majority class

Cohen's Kappa

- As shown below, 77.88% of the customers in data did not miss their payment, so $p_e = 0.7788$
  - Unfortunately, our model's classification accuracy was also 77.88%, so $p_o = 0.7788$
- So, $\kappa = \frac{0.7788 - 0.778}{1 - 0.7788} = 0$, which suggests this model is ineffective

```r
table(fr$MISSED_PAYMENT)
```

```
##
##     0     1
## 23364  6636
```

```r
23364/(23364 + 6636)
```

```
## [1] 0.7788
```

A More Complex Model

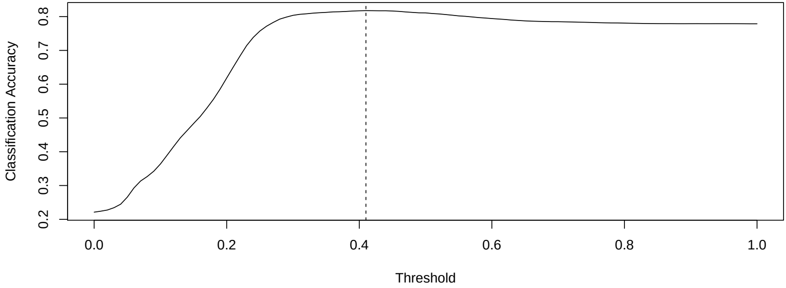
Let's now consider a model that uses every predictor in the dataset:

```r
m <- glm(MISSED_PAYMENT ~ LIMIT_BAL + SEX + EDUCATION + MARRIAGE + AGE +
           PAY_0 + PAY_2 + PAY_3 + PAY_4 + PAY_5 + PAY_6 +
           BILL_AMT1 + BILL_AMT2 + BILL_AMT3 + BILL_AMT4 + BILL_AMT5 + BILL_AMT6 +
           PAY_AMT1 + PAY_AMT2 + PAY_AMT3 + PAY_AMT4 + PAY_AMT5 + PAY_AMT5,
           data = fr, family = "binomial")
pred_probs <- predict(m, type = "response")
t = 0.5
pred_class <- ifelse(pred_probs > t, 1, 0)
sum(pred_class == fr$MISSED_PAYMENT)/nrow(fr)
```

```
## [1] 0.8110667
```

Finding a Threshold

We can further refine things by finding an optimal probability threshold:




```r
data.frame(threshold = tseq[which.max(acc)], accuracy = acc[which.max(acc)])
```

```
##   threshold accuracy
## 1      0.41   0.8177
```

A More Complex Model

- Using a probability threshold of 0.41, this model can predict missed payments 81.77% accuracy
  - The corresponding Cohen's kappa is $\frac{0.8177-0.7788}{1-0.7788} = 0.18$

A More Complex Model

- Using a probability threshold of 0.41, this model can predict missed payments 81.77% accuracy
  - The corresponding Cohen's kappa is $\frac{0.8177-0.7788}{1-0.7788} = 0.18$
  - This model has *some predictive ability*, though not a whole lot (recall κ ranges from 0 to 1)

A More Complex Model

- Using a probability threshold of 0.41, this model can predict missed payments 81.77% accuracy
  - The corresponding Cohen's kappa is $\frac{0.8177-0.7788}{1-0.7788} = 0.18$
  - This model has *some predictive ability*, though not a whole lot (recall κ ranges from 0 to 1)
- Unfortunately, we're also ignoring that the model used a large number of explanatory variables and *in-sample* accuracy
  - How would you expect its *out-of-sample* accuracy to compare?

Is the Model Overfit?

Cross-validation can be used to estimate out-of-sample performance:

```
set.seed(123)
fold_id <- sample(rep(1:5, length.out = nrow(fr)), size = nrow(fr))
preds <- numeric(nrow(fr))
for(k in 1:5){

  ## Subset the data
  train <- fr[fold_id != k, ]
  test <- fr[fold_id == k, ]

  ## Fit models on the data
  m <- glm(MISSED_PAYMENT ~ LIMIT_BAL + SEX + EDUCATION + MARRIAGE + AGE +
           PAY_0 + PAY_2 + PAY_3 + PAY_4 + PAY_5 + PAY_6 +
           BILL_AMT1 + BILL_AMT2 + BILL_AMT3 + BILL_AMT4 + BILL_AMT5 + BILL_AMT6 +
           PAY_AMT1 + PAY_AMT2 + PAY_AMT3 + PAY_AMT4 + PAY_AMT5 + PAY_AMT5,
           data = train, family = "binomial")

  ## Store predictions
  preds[fold_id == k] <- predict(m, newdata = test, type = "response")
}

## Out of sample accuracy
pred_class <- ifelse(preds >= 0.41, 1, 0)
sum(pred_class == fr$MISSED_PAYMENT)/nrow(fr)
```

```
## [1] 0.8171
```

Does the model appear to be overfit?

Misclassification Errors

- As a final consideration, we might ask: “is it equally bad for the model to predict”miss" for someone who makes their payment as it is for it to predict “make” for someone who misses their payment?"

Misclassification Errors

- As a final consideration, we might ask: “is it equally bad for the model to predict”miss" for someone who makes their payment as it is for it to predict “make” for someone who misses their payment?"
- One way to explore the frequencies of each type of misclassification is a **confusion matrix**:

```
table(fr$MISSED_PAYMENT, pred_class)
```

```
##    pred_class
##         0     1
##   0 22042  1322
##   1  4165  2471
```

Sensitivity and Specificity

The likelihood of each type of misclassification is captured in two probabilities known as **sensitivity** and **specificity**:

1) Sensitivity - The probability of a *true positive*, or a case who missed their payment being classified as “missed” or “1”

Sensitivity and Specificity

The likelihood of each type of misclassification is captured in two probabilities known as **sensitivity** and **specificity**:

1) Sensitivity - The probability of a *true positive*, or a case who missed their payment being classified as "missed" or "1"
2) Specificity - The probability of a *true negative*, or a case who made their payment being classified as "made" or "0"

Sensitivity and Specificity

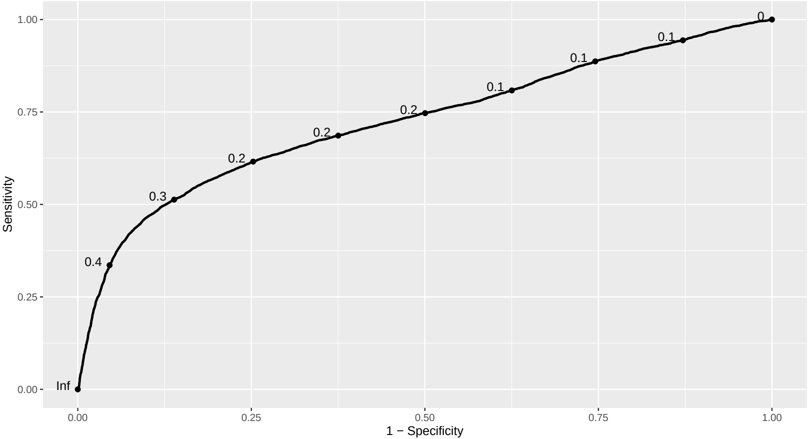
The likelihood of each type of misclassification is captured in two probabilities known as **sensitivity** and **specificity**:

1) Sensitivity - The probability of a *true positive*, or a case who missed their payment being classified as "missed" or "1"
2) Specificity - The probability of a *true negative*, or a case who made their payment being classified as "made" or "0"

- Because the confusion matrix will change in response to the classification threshold, t, so will the sensitivity and specificity

ROC Analysis

The trade-off between sensitivity and specificity, taken across a variety of decision thresholds, can be expressed via a Receiver Operating Characteristic (ROC) curve:



AUC

- The area under the ROC curve (AUC) provides an overall model summary
  - A model with no predictive value will have an AUC of 0.5 (imagine this as a straight line connecting (0,0) and (1,1))
  - What AUC value will a *perfect model* achieve?

```r
## AUC for our missed payment model
roc_plot <- ggplot(df, aes(d = class, m = pi)) + geom_roc()
calc_auc(roc_plot)
```

```
##   PANEL group       AUC
## 1     1    -1 0.7241173
```

Closing Remarks

- We've now covered a few different ways to quantify the performance of a logistic regression model:

Metric	Easily understood	Adjusts for imbalance	Invariant to t
Accuracy	YES	NO	NO
Cohen's Kappa	SORT OF	YES	NO
AUC	NO	YES	YES

- Depending upon the details of your application, one of these metrics might be preferable.
- Additionally, if overfitting is a concern, you should use cross-validation to estimate the out-of-sample version of your metric of choice.