Lesson 9

Clark

Today we're going to talk a bit more about some modifications we can make to our binary regression model. Recall we refer to our binary regression model as a logistic regression model if our link function is the logistic function. One reason we might use the logistic function is that it is the canonical link.

However, that's not the only motivation. Let's consider a different framework. Let's say we are interested in whether a Cadet passes the IOCT or not. Now, the IOCT is quite difficult, say d, and perhaps a Cadet only passes the IOCT if their ability T is greater than d. However, we don't really know a Cadet's true ability, so we can consider T as a random variable. Therefore the probability that you fail the IOCT is given by:

If we allow the probability distribution of T to follow the logistic distribution, we have:

Setting p from above equal to $F(y)$ we can solve for:

Thus, using the logit link is equivalent to assuming that there is a latent variable (i.e. a variable that we cannot observe) that follows a logistic distribution. If we *could* observe T then if it was bigger than d we would pass, but because we cannot observe it we have to use a probability distribution.

Now, if we use a different distribution for our latent variable, it turns out we get different link functions. For instance, if instead we assume our latent variable follows a Normal distribution, we end up with:

This is called the *probit* link function. So, what is the practical difference? Really not much. If we take a look at the plots we get:

The real thing to note is that our interpretation of our covariates differ when I change link functions. When we used a logistic link our covariates could be understood as a change in the log-odds, or changing the odds by a multiplicative factor of $\exp(\beta_j)$. Probit is not so nice

There's also a complimentary log-log that might get used. But, really, if we look at page 70 in Faraway we can see that the actual changes to the curves aren't very significant. The one thing that does hold true, though, is that all of these functions are monotonic increasing, so a positive β, regardless of link function leads to an increasing probability of success and a negative covariate value leads to a decreasing probabilty of success. Really, I feel very little use in any link function outisde of the logit, but there's nothing wrong with using the other ones as long as you keep in mind that your interpretation of your covariates may not be clean.

I think the last thing to talk about is how do we evaluate our binary regression models. Certainly, if our models are nested we could:

Or even if they aren't, there's nothing to stop us from calculating AIC or BIC. However, there are some other ways that people typically compare models for binary outcomes.

The first is the the (appropriately named) Confusion matrix. This is a matrix of all of our outcomes:

This is typically done to compare different models. Let's take a look at our heart data from last class

```
library(tidyverse)
library(faraway)
data(wcgs)
wcgs <- wcgs%>%drop_na()
one_glm <- glm(chd~cigs+weight+chol+age+arcus+
                 behave,family=binomial(link="logit"),
               data=wcgs)

another_glm <- glm(chd~cigs*weight+chol+age,family=binomial(link="logit"),
data=wcgs)
```

Here our models are nested, so we could run:

```
anova(one_glm,another_glm, test="Chi")
```

```
## Analysis of Deviance Table
##
## Model 1: chd ~ cigs + weight + chol + age + arcus + behave
## Model 2: chd ~ cigs * weight + chol + age
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1      3131     1586.5
## 2      3134     1611.3 -3  -24.847 1.662e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Or we could do:

```
print(paste0("First AIC - ",format(AIC(one_glm),digits=5)," Another AIC - ",format(AIC(another_glm),dig
```

```
## [1] "First AIC - 1604.5 Another AIC - 1623.3"
```

Or, we could look at the confusion matrices

```
library(caret)
full_pred <- predict(one_glm, type="response")
preds <- ifelse(full_pred>.5,"yes","no")
my_cm <- confusionMatrix(data = as.factor(preds), reference = as.factor(wcgs$chd))
my_cm$table
```

```
##           Reference
## Prediction   no  yes
##        no  2877  253
##        yes    8    2
```

So, not exactly good here… So, what do we think the issue is?

```
preds <- ifelse(full_pred>.2,"yes","no")
my_cm <- confusionMatrix(data = as.factor(preds), reference = as.factor(wcgs$chd))
my_cm$table
```

```
##           Reference
## Prediction   no  yes
##        no  2717  201
##        yes  168   54
```

Perhaps a bit better. However, we've got an issue. What is the problem with how we've been doing business here?

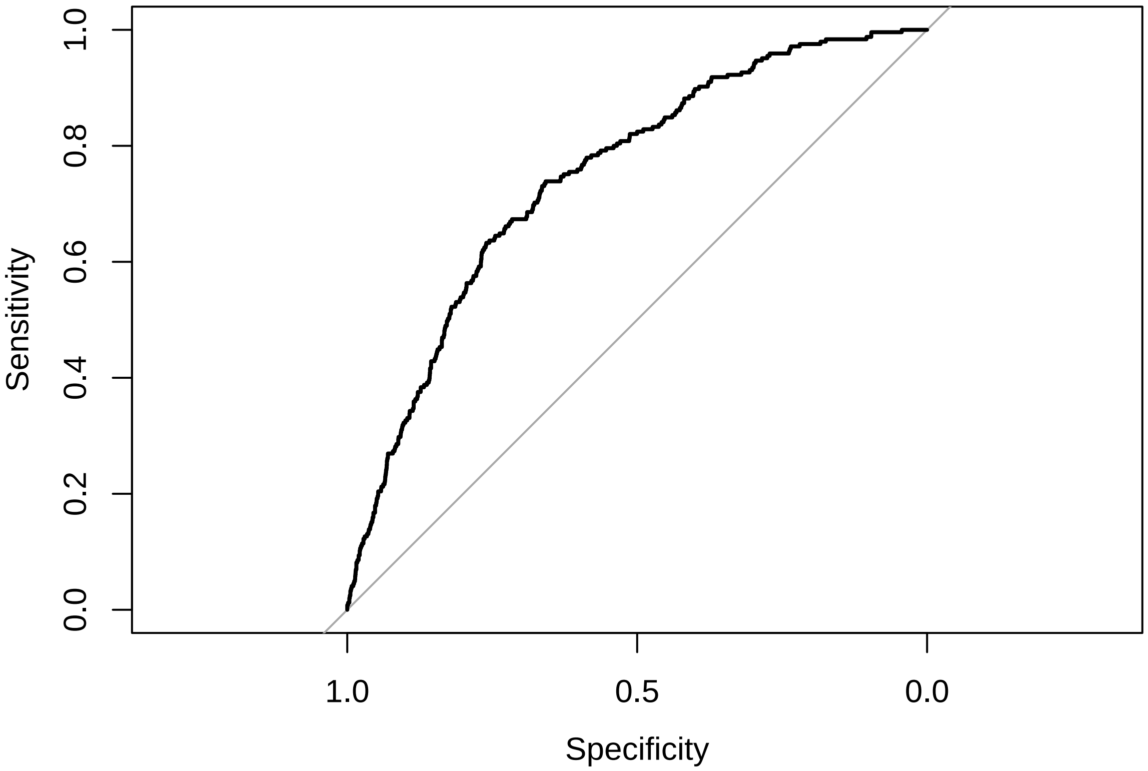
```
set.seed(1)
test_index <- sample(1:nrow(wcgs), 100, replace = FALSE)
train_data <- wcgs[-test_index,]
test_data <- wcgs[test_index,]

train_glm <- glm(chd~cigs+weight+chol+age+arcus+
                 behave,family=binomial(link="logit"),
               data=train_data)
test_pred <- predict(train_glm,test_data, type="response")
preds_class <- ifelse(test_pred>.2,"yes","no")
my_cm <- confusionMatrix(data = as.factor(preds_class), reference = as.factor(test_data$chd))
my_cm$table
```

```
##           Reference
## Prediction no yes
##        no  86   8
##        yes  4   2
```

This whole business of picking a cut-off point may seem a bit arbitrary. Perhaps we want to know how well a model performs across a wide range of classification thresholds. An ROC (receiver operating characteristic) curve shows the performance of a classificaiton model at all classificaiton thresholds.

```
library(pROC)
myroc <- roc(as.factor(train_data$chd), predict(train_glm, type="response"))
plot(myroc)
```



The ROC curve plots the true negative rate vs the true positive rate. It picks a bunch of cut off points and then computes the sensitivity (number correctly identified 1s)/(total number of observed 1s) ad the specificity (number correctly identified 0s)/(total number of observed 0s)

```
myroc$thresholds[1500]
```

```
## [1] 0.05699083
```

```
myroc$sensitivities[1500]
```

```
## [1] 0.8081633
```

```
myroc$specificities[1500]
```

```
## [1] 0.5202147
```

So, if you are good with a sensitivity of 80% and a specificity of 52% pick 0.057 as your threshold

```
preds_class <- ifelse(test_pred>.057,"yes","no")
my_cm <- confusionMatrix(data = as.factor(preds_class), reference = as.factor(test_data$chd))
my_cm$table
```

```
##           Reference
## Prediction no yes
##        no  50   1
##        yes 40   9
```

Not exactly, but remember, this was computed on our training set.

If I want to compare two different models, I could compute the area underneath the ROC curve (AUC). For instance:

```r
train1_glm <- glm(chd~cigs+weight+chol+age+arcus+
                 behave,family=binomial(link="logit"),
               data=train_data)


train2_glm <- glm(chd~cigs*weight+chol*age+arcus+
                 behave,family=binomial(link="logit"),
               data=train_data)


myroc1 <- roc(as.factor(test_data$chd), predict(train1_glm,test_data, type="response"))

myroc2 <- roc(as.factor(test_data$chd), predict(train2_glm, test_data, type="response"))

myroc1$auc
```

```
## Area under the curve: 0.7844
```

```r
myroc2$auc
```

```
## Area under the curve: 0.79
```

So, by this statistic the second model would be preferred. HOWEVER, this does NOT take into account model complexity like AIC does.

```r
AIC(train1_glm)
```

```
## [1] 1546.93
```

```r
AIC(train2_glm)
```

```
## [1] 1548.92
```

Though, because I am finding AUC using training/testing split, it would penalize overly complex models by ensuring we are not overfitting our data. In fact, we could say that AUC measures a model's predictive performance while AIC or BIC estimates the model's predictive performance.

However, unlike what you will read some places, this doesn't mean we should just abandon AIC or BIC. This ASSUMES that our testing data is, indeed, representative! If our testing data is crap, then we aren't guaranteed of really anything. Unless we are going to go through the act of assessing the characteristics of our testing data we likely have just switched from one set of assumptions to another.