Logistic Regression

2019-03-26

In this set of notes, you will learn how logistic regression models can be used to model dichotomous outcome variables.

Dataset and Research Question

In this set of notes, we will use data from the file *graduation.csv* (see the data codebook here). These data include student-level attributes for $n = 2344$ randomly sampled students who were first-time, full-time freshman from the 2002 cohort at a large, midwestern research university.

```
# Load libraries
library(AICcmodavg)
library(broom)
library(corrr)
library(dplyr)
library(ggplot2)
library(readr)
library(sm)
library(tidyr)

# Read in data
grad = read_csv(file = "~/Documents/github/epsy-8252/data/graduation.csv")

# View data
head(grad)
```

```
# A tibble: 6 x 6
  degree   act scholarship    ap firstgen nontrad
   <dbl> <dbl>       <dbl> <dbl>    <dbl>   <dbl>
1      1  21           0       0        0       0
2      1  19.0         0       0        0       0
3      1  27           0       0        1       0
4      1  25           0.5     0        1       0
5      0  28           0      17        1       0
6      1  21           0       0        0       1
```

We will use these data to explore predictors of college graduation.

In the last set of notes, we saw that using the linear probability model leads to direct violations of the linear model's assumptions. If that isn't problematic enough, it is possible to run into severe issues when we make predictions. For example, given the constant effect of X in these models it is possible to have an X value that results in a predicted proportion that is either greater than 1 or less than 0. This is a problem since proportions are constrained to the range of $[0,\ 1]$.

Before we consider any alternative models, let's actually examine the empirical proportions of students who graduate at different ACT scores.

```
graduates = grad %>%
  group_by(act, degree) %>%
  summarize( N = n() ) %>%
  mutate( Prop = N / sum (N) ) %>%
  filter(degree == 1) %>%
  ungroup() #Makes the resulting tibble regular

# View data
head(graduates, 10)
```

```
# A tibble: 10 x 4
     act degree     N  Prop
   <dbl>  <dbl> <int> <dbl>
 1  11         1     1 0.25
 2  13         1     4 0.5
 3  14         1     6 0.5
 4  15         1    13 0.448
 5  16         1     6 0.24
 6  17         1    18 0.429
 7  18         1    26 0.531
 8  19         1    49 0.681
 9  19.0       1     1 1
10  20         1    74 0.679
```

```
# Plot proportions
ggplot(data = graduates, aes(x = act, y = Prop)) +
  geom_point() +
  geom_smooth(se = FALSE) +
  theme_bw() +
  xlab("ACT score") +
  ylab("Proportion of graduates")
```

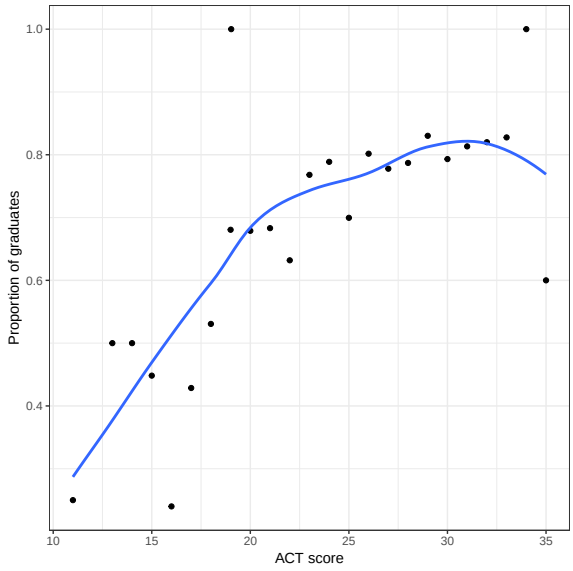
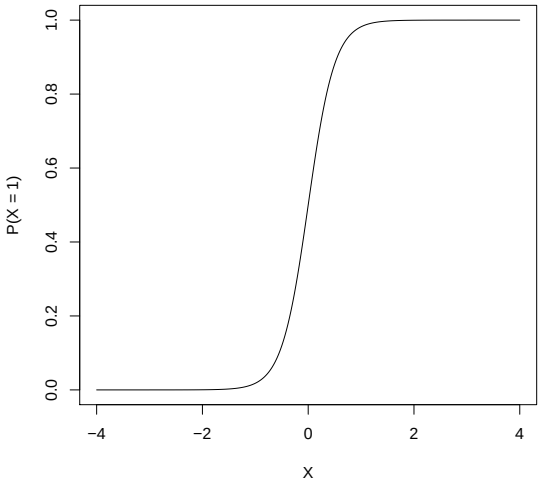


Figure 1: The loess smoother suggests that the proportion of students who graduate is a non-linear function of ACT scores.

Alternative Models to the Linear Probability Model

Many of the non-linear models that are typically used to model dichotomous outcome data are "S"-shaped models. Below is a plot of one-such "S"-shaped model.



The non-linear "S"-shaped model has many attractive features. First, the predicted Y values are bounded between 0 and 1. Furthermore, as X gets smaller, the proportion of $Y = 1$, approaches 0 at a slower rate. Similarly, as X gets larger, the proportion of $Y = 1$, approaches 1 at a slower rate. Lastly, this model curve is monotonic; smaller values of X are associated with smaller proportions of $Y = 1$ (or if the "S" were backwards, larger values of X would be associated with smaller proportions of $Y = 1$). The key is that there are no bends in the curve; it is always growing or always decreasing.

In our graduation example, the empirical data maps well to this curve. Higher ACT scores are associated with a higher proportion of students who graduate (monotonic). The effect of ACT, however, is not constant, and seems to diminish at higher ACT scores. Lastly, we want to bound the proportion at every ACT score to lie between 0 and 1.

How do we fit such an "S"-shaped curve? We apply a transformation function, call it Λ, to the predicted values. Mathematically,

$$\Lambda(\pi_i) = \Lambda\Big[\beta_0 + \beta_1(X)\Big]$$

The specific transformation function used is any mathematical function that can fit the criteria we had before (monotonic, nonlinear, maps to $[0,\ 1]$ space). There are several mathematical functions that do this. One common function that meets these specifications is the *logistic function*. Mathematically, the logistic function is

$$\Lambda(w) = \frac{1}{1 + e^{-w}}$$

where w is the value fed into the logistic function. For example, to logistically transform $w = 3$, we use

$$\begin{aligned}\Lambda(3) &= \frac{1}{1 + e^{-3}} \\ &= 0.953\end{aligned}$$

Below we show how to transform many such values using R.

```
example = data.frame(
  w = seq(from = -4, to = 4, by = 0.01)  # Set up values
  ) %>%
  mutate(
    Lambda = 1 / (1 + exp(-w))  # Transform using logistic function
  )

# View data
head(example)
```

```
      w Lambda
1 -4.00 0.0180
2 -3.99 0.0182
3 -3.98 0.0183
4 -3.97 0.0185
5 -3.96 0.0187
6 -3.95 0.0189
```

```
# Plot the results
ggplot(data = example, aes(x = w, y = Lambda)) +
  geom_line() +
  theme_bw()
```

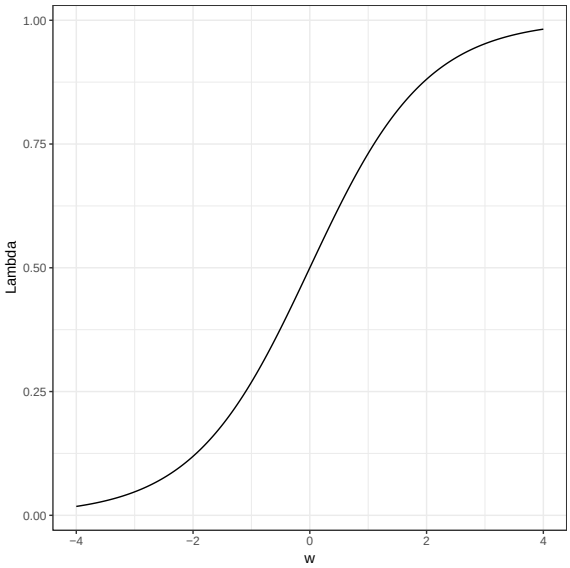


Figure 2: Plot of the logistically transformed values for a sequence of values from -4 to 4.

You can see that by using this transformation we get a monotonic "S"-shaped curve. Now try substituting a really large value of w into the function. This gives an asymptote at 1. Also substitute a really "large"" negative value in for w. This gives an asymptote at 0. So this function also bounds the output between 0 and 1.

How does this work in a regression? There, we are transforming the *predicted values*, the π_i values, which we express as a function of the predictors:

$$\begin{aligned}\Lambda(\hat{\pi}_i) &= \Lambda\Big[\beta_0 + \beta_1(X_i)\Big] \\ &= \frac{1}{1 + e^{-\left[\beta_0 + \beta_1(X_i)\right]}}\end{aligned}$$

Since we took a linear model ($\beta_0 + \beta_1(X_i)$) and applied a logistic transformation, the resulting model is the *linear logistic model* or more simply, the *logistic model*.

Re-Expressing a Logistic Transformation

The logistic model expresses the proportion of 1s (π_i) as a function of the predictor X. It can be mathematically expressed as

$$\pi_i = \frac{1}{1 + e^{-\left[\beta_0 + \beta_1(X_i)\right]}}$$

We can re-express this using algebra and rules of logarithms.

$$\begin{aligned}
\pi_i &= \frac{1}{1+e^{-\left[\beta_0+\beta_1(X_i)\right]}} \\
\pi_i \times (1+e^{-\left[\beta_0+\beta_1(X_i)\right]}) &= 1 \\
\pi_i + \pi_i(e^{-\left[\beta_0+\beta_1(X_i)\right]}) &= 1 \\
\pi_i(e^{-\left[\beta_0+\beta_1(X_i)\right]}) &= 1-\pi_i \\
e^{-\left[\beta_0+\beta_1(X_i)\right]} &= \frac{1-\pi_i}{\pi_i} \\
e^{\left[\beta_0+\beta_1(X_i)\right]} &= \frac{\pi_i}{1-\pi_i} \\
\ln\left(e^{\left[\beta_0+\beta_1(X_i)\right]}\right) &= \ln\left(\frac{\pi_i}{1-\pi_i}\right) \\
\beta_0+\beta_1(X_i) &= \ln\left(\frac{\pi_i}{1-\pi_i}\right)
\end{aligned}$$

Or,

$$\ln\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0+\beta_1(X_i)$$

The logistic model expresses the natural logarithm of $\frac{\pi_i}{1-\pi_i}$ as a linear function of X. Note that there is no error term on this model. This is because the model is for the mean structure only (the proportions), we are not modeling the actual Y_i values with the logistic regression model.

Log-Odds or Logits

The ratio that we are taking the logarithm of, $\frac{\pi_i}{1-\pi_i}$, is referred to as *odds*. Odds are the ratio of two probabilities. Namely the chance an event occurs (π_i) versus the chance that same event does not occur ($1-\pi_i$). As such, it gives the *relative chance* of that event occurring. To understand this better, we will look at a couple examples.

Let's assume that the probability of getting an "A" in a course is 0.7. Then we know the probability of NOT getting an "A" in that course is 0.3. The odds of getting an "A" are then

$$\text{Odds} = \frac{0.7}{0.3} = 2.33$$

That the probability of getting an "A" in the class is 2.33 times as likely as NOT getting an "A" in the class. This is the relative probability of getting an "A".

As another example, Fivethirtyeight.com computed the probability that a Canadian hockey team would win the Stanley Cup in 2018 as 0.17. The odds of a Canadian team winning the Stanley Cup is then

$$\text{Odds} = \frac{0.17}{0.83} = 0.21$$

The probability that a Canadian team wins the Stanley Cup is 0.21 times as likely as a Canadian team NOT winning the Stanley Cup. (Odds less than 1 indicate that is is more likely for an event NOT to occur than to occur. Invert the fraction to compute how much more like the event is not to occur.)

In the logistic model, we are predicting the log-odds (also referred to as the *logit*. When we get these values, we typically transform the logits to odds by inverting the log-transformation (take e to that power.)

Binomially Distributed Errors

The logistic transformation fixed two problems: (1) the non-linearity in the conditional mean function, and (2) bounding any predicted values between 0 and 1. However, just fitting this transformation does not fix the problem of non-normality. Remember from the previous notes we learned that at each X_i there were only two potential values for Y_i; 0 or 1. Rather than use a normal (or Gaussian) distribution to model the conditional distribution of Y_i, we will use the *binomial distribution*. The binomial distribution is a discrete probability distribution that gives the probability of obtaining exactly n successes out of N Bernoulli trials (where the result of each Bernoulli trial is true with probability π and false with probability $1 - \pi$). This is appropriate since at each value of X we can posit N Bernoulli trials, n of which are 1 (successes).

Fitting the Binomial Logistic Model in R

The syntax to fit the logistic model using `glm()` is

$$\texttt{glm}(\text{y} \sim 1 + \text{x}, \texttt{data} = \text{dataframe}, \texttt{family} = \texttt{binomial}(\texttt{link} = \texttt{logit})$$

The formula depicting the model and the `data=` arguments are specified in the same manner as in the `lm()` function. Since the model is a generalized model, we need to specify the distribution for the conditional Y_i values (binomial) and the link function (logit) via the `family=` argument.

For our example,

```
glm.1 = glm(degree ~ 1 + act, data = grad, family = binomial(link = "logit"))
```

The output of the model can be printed using `summary()`.

```
summary(glm.1)
```

```
Call:
glm(formula = degree ~ 1 + act, family = binomial(link = "logit"),
    data = grad)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.127  -1.224   0.697   0.802   1.364

Coefficients:
            Estimate Std. Error z value             Pr(>|z|)
(Intercept)  -1.6114     0.2827   -5.70          0.000000012 ***
act           0.1076     0.0116    9.25 < 0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2722.5  on 2343  degrees of freedom
Residual deviance: 2633.2  on 2342  degrees of freedom
AIC: 2637

Number of Fisher Scoring iterations: 4
```

The fitted equation for the model is

$$\ln\left(\frac{\hat{\pi}_i}{1-\hat{\pi}_i}\right) = -1.61 + 0.11(\text{ACT Score}_i)$$

We interpret the coefficients in the same manner as we interpret coefficients from a linear model, with the caveat that the outcome is now in log-odds (or logits):

- For students with an ACT score of 0, their predicted log-odd of graduating are -1.61.
- Each one-point difference in ACT score is associated with a difference of 0.11in the predicted log-odds of graduating, on average.

For better interpretations, we can back-transform log-odds to odds. This is typically a better metric for interpretation of the coefficients. To back-transform to odds, we exponentiate both sides of the fitted equation and use the rules of exponents to simplify:

$$\begin{aligned} e^{\ln\left(\frac{\hat{\pi}_i}{1-\hat{\pi}_i}\right)} &= e^{-1.61+0.11(\text{ACT Score}_i)} \\ \frac{\hat{\pi}_i}{1-\hat{\pi}_i} &= e^{-1.61} \times e^{0.11(\text{ACT Score}_i)} \end{aligned}$$

When ACT score = 0, the predicted odds are

$$\begin{aligned} \frac{\hat{\pi}_i}{1-\hat{\pi}_i} &= e^{-1.61} \times e^{0.11(0)} \\ &= e^{-1.61} \times 1 \\ &= e^{-1.61} \\ &= 0.2 \end{aligned}$$

For students with an ACT score of 0, their odds of graduating is 0.2. That is, for these students, the probability of graduating is 0.2 times that of not graduating. (It is far more likey these students will not graduate.)

To interpret the effect of ACT on the odds of graduating, we will compare the odds of graduating for students that have ACT score that differ by one point. Say ACT = 0 and ACT = 1.

We already know the predicted odds for students with ACT = 0, namely $e^{-1.61}$. For students with an ACT of 1, their predicted odds of graduating are

$$\begin{aligned} \frac{\hat{\pi}_i}{1-\hat{\pi}_i} &= e^{-1.61} \times e^{0.11(1)} \\ &= e^{-1.61} \times e^{0.11} \end{aligned}$$

These students odds of graduating are $e^{0.11}$ times greater than students with an ACT score of 0. Moreover, this increase in the odds, on average, is the case for every one-point difference in ACT score. In general,

- The predicted odds for $X = 0$ are $e^{\hat{\beta}_0}$.
- Each one-unit difference in X is associated with a $e^{\hat{\beta}_1}$ times increase (decrease) in the odds.

We can obtain these values in R by using the `coef()` function to obtain the fitted model's coefficients and then exponentiating them using the `exp()` function.

```
exp(coef(glm.1))
```

```
(Intercept)         act
       0.20        1.11
```

From these values, we interpret the coefficients in the odds metric as

- The predicted odds of graduating for students with an ACT score of 0 are 0.20.
- Each one-unit difference in ACT score is associated with 1.11 times greater odds of graduating.

Plotting the Results from the Fitted Model

To even further understand and interpret the fitted model, we can plot the fitted values for a range of ACT scores. Typically the proportion of students graduating ($\hat{\pi}_i$) is what should be plotted, as that is what we were initially modeling. This process is exactly the same as the process of plotting fitted model results for any of the other models we have worked with. The only difference is that when we use the `predict()` function to get the fitted values, we have to specify that we want the $\hat{\pi}_i$ values (the default is logits). We do this by including the argument `type="response"`. Below we plot the results from our fitted logistic model.

```
# Create the data to plot
plotData = crossing(
  act = seq(from = 10, to = 36, by = 1)
  ) %>%
  mutate(
    pi_hat = predict(glm.1, newdata = ., type = "response")
  )

# Plot the data
ggplot(data = plotData, aes(x = act, y = pi_hat)) +
  geom_line() +
  theme_bw() +
  xlab("ACT score") +
  ylab("Predicted probability of graduating") +
  ylim(0, 1)
```

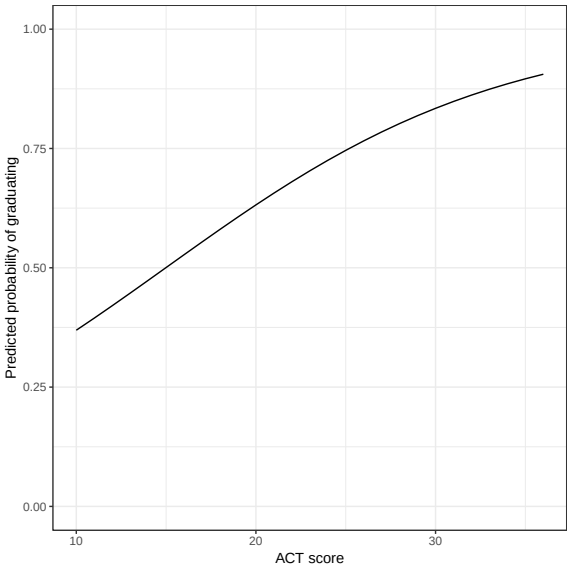


Figure 3: Predicted probability of graduating college as a function of ACT score.

The monotonic increase in the curve indicates the positive effect of ACT score on the probability of graduating. (Note that we typically interpret π_i as probability rather than proportion when interpreting logistic models.) The magnitude of this effect depends on ACT score. For lower ACT scores there is a larger effect of ACT score on the probability of graduating than for higher ACT scores.

Model-Level Summaries

The `summary()` output for the GLM model also included model-level information. For the model we fitted, the model-level information was

```
    Null deviance: 2722.5  on 2343  degrees of freedom
Residual deviance: 2633.2  on 2342  degrees of freedom
AIC: 2637

Number of Fisher Scoring iterations: 4
```

The metric of measuring residual fit is the deviance (remember the deviance was $-2\times$ log-likelihood). The *null deviance* is the residual deviance from fitting the intercept-only model.

```
# Fit intercept-only model
glm.0 = glm(degree ~ 1, data = grad, family = binomial(link = "logit"))

# Compute deviance
-2 * logLik(glm.0)[[1]]
```

```
[1] 2723
```

The *residual deviance* is the residual deviance from fitting whichever model was fitted, in our case the model that used ACT score as a predictor.

```
-2 * logLik(glm.1)[[1]]
```

```
[1] 2633
```

Recall that deviance is akin to the sum of squared residuals (SSE) in conventional linear regression; smaller values indicate less error. In our case, the model that includes ACT score as a predictor has less error than the intercept only model; its deviance is 90 less than the intercept-only model.

There are two ways to determine whether this decrease in deviance is statistically significant. The first is to examine the significance of the ACT predictor in the fitted model. Since that is the only predictor included above-and-beyond the intercept, the p-value associated with it indicates whether the ACT predictor is statistically relevant.

The second method to test the improvement in deviance is a test of nested models. Since the intercept-only model is nested in the model that includes ACT as a predictor, we can use a *Likelihood Ratio Test* to examine this. To do so, we use the `anova()` function with the added argument `test="LRT"`.

```
anova(glm.0, glm.1, test = "LRT")
```

```
Analysis of Deviance Table

Model 1: degree ~ 1
Model 2: degree ~ 1 + act
  Resid. Df Resid. Dev Df Deviance            Pr(>Chi)
1      2343       2723
2      2342       2633  1     89.3 <0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis of this test is that there is NO improvement in the deviance. The results of this test, $\chi^2(1) = 89.3$, $p < .001$, indicate that we should reject the null hypothesis. The more complex model has significantly less error than the intercept-only model and should be adopted.

We could have also used information criteria for making decisions about effects.

```
myAIC = aictab(
  cand.set = list(glm.0, glm.1),
  modnames = c("Intercept-only", "ACT score")
)

myAIC
```

```
Model selection based on AICc:

               K AICc Delta_AICc AICcWt Cum.Wt    LL
ACT score      2 2637        0.0      1      1 -1317
Intercept-only 1 2725       87.3      0      1 -1361
```

Here, given the data and the candidate set of models, there is overwhelming evidence to support the model that includes ACT score.

Including Covariates

Including covariates in the logistic model is done the same way as for `lm()` models. For example, say we wanted to examine the effect of ACT score on probability of graduating, after controlling for whether or not a student was first generation college student. We fit that model as

```
# Fit model
glm.2 = glm(degree ~ 1 + act + firstgen, data = grad, family = binomial(link = "logit"))

# Obtain summary output
summary(glm.2)
```

```
Call:
glm(formula = degree ~ 1 + act + firstgen, family = binomial(link = "logit"),
    data = grad)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.113  -1.185   0.660   0.796   1.401

Coefficients:
            Estimate Std. Error z value         Pr(>|z|)
(Intercept)  -1.4810     0.2847   -5.20 0.00000019747419 ***
act           0.0882     0.0123    7.19 0.00000000000066 ***
firstgen      0.5155     0.1044    4.94 0.00000079745988 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2722.5  on 2343  degrees of freedom
Residual deviance: 2609.2  on 2341  degrees of freedom
AIC: 2615

Number of Fisher Scoring iterations: 4
```

The fitted equation is

$$\ln\left(\frac{\hat{\pi}_i}{1-\hat{\pi}_i}\right) = -1.48 + 0.09(\text{ACT Score}_i) + 0.52(\text{First Generation}_i)$$

Using the logit/log-odds metric, we interpret the coefficients as:

- For students who are not first generation college students with an ACT score of 0, the predicted log-odds of graduating are -1.48, on average.
- Each one-point difference in ACT score is associated with a difference of 0.09 in the predicted log-odds of graduating, on average, after controlling for whether o not the students are first generation college students.
- First generation college students, on average, have a predicted log-odds of graduating that is 0.52 higher than students who are not first generation students, after controlling for differences in ACT scores.

If we back-transform to odds,

```
exp(coef(glm.2))
```

```
(Intercept)         act    firstgen
      0.227       1.092       1.674
```

The interpretations are:

- For students with an ACT score of 0 who are not first generation college students, the predicted odds of graduating are 0.23, on average.
- Each one-point difference in ACT score is associated with improving the odds of graduating 1.09 times, on average, after controlling for whether o not the students are first generation college students.
- First generation college students, on average, predicted odds of graduating are 1.67 times that of students who are not first generation students, after controlling for differences in ACT scores.

We can also plot the predicted probability of graduating to aid interpretation.

```
# Create data to plot
plotData = crossing(
  act = seq(from = 10, to = 36, by = 1),
  firstgen = c(0, 1)
  ) %>%
  mutate(
    pi_hat = predict(glm.2, newdata = ., type = "response"),
    firstgen = factor(firstgen,
                      levels = c(0, 1),
                      labels = c("Non First Generation Students", "First Generation Students")
                      )
  )


# Plot the data
ggplot(data = plotData, aes(x = act, y = pi_hat, color = firstgen)) +
  geom_line() +
  theme_bw() +
  xlab("ACT score") +
  ylab("Predicted probability of graduating") +
  ylim(0, 1) +
  scale_color_brewer(name = "", palette = "Set1")
```

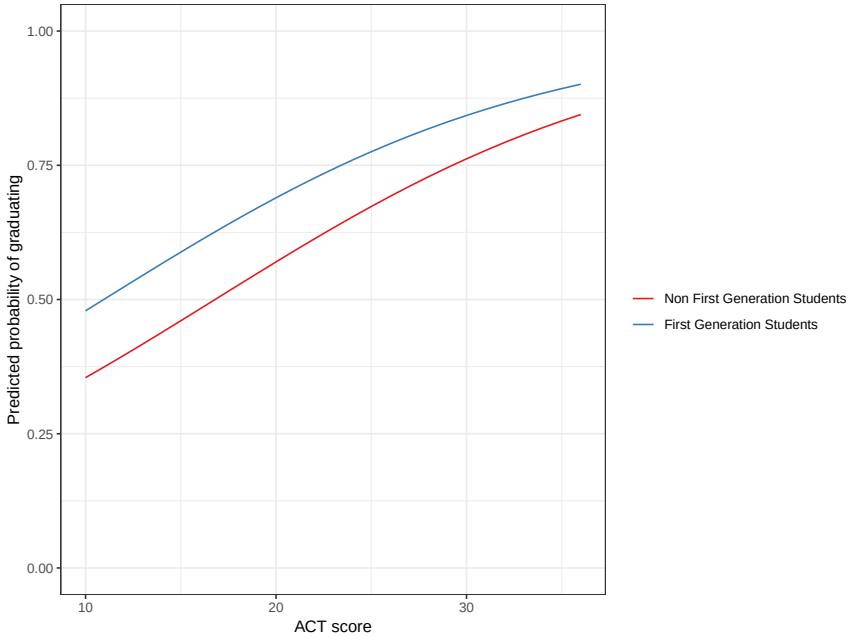


Figure 4: Predicted probability of graduating college as a function of ACT score for first and non-first generation students.

Other Resources

In addition to the notes and what we cover in class, there many other resources for learning about using binomial logistic regression models for analyzing binary data. Here are some resources that may be helpful in that endeavor:

- Section 3.2.1: *The Binomial and Bernoulli Distributions* in Fox (2009) [Required Textbook]