Logistic Regression

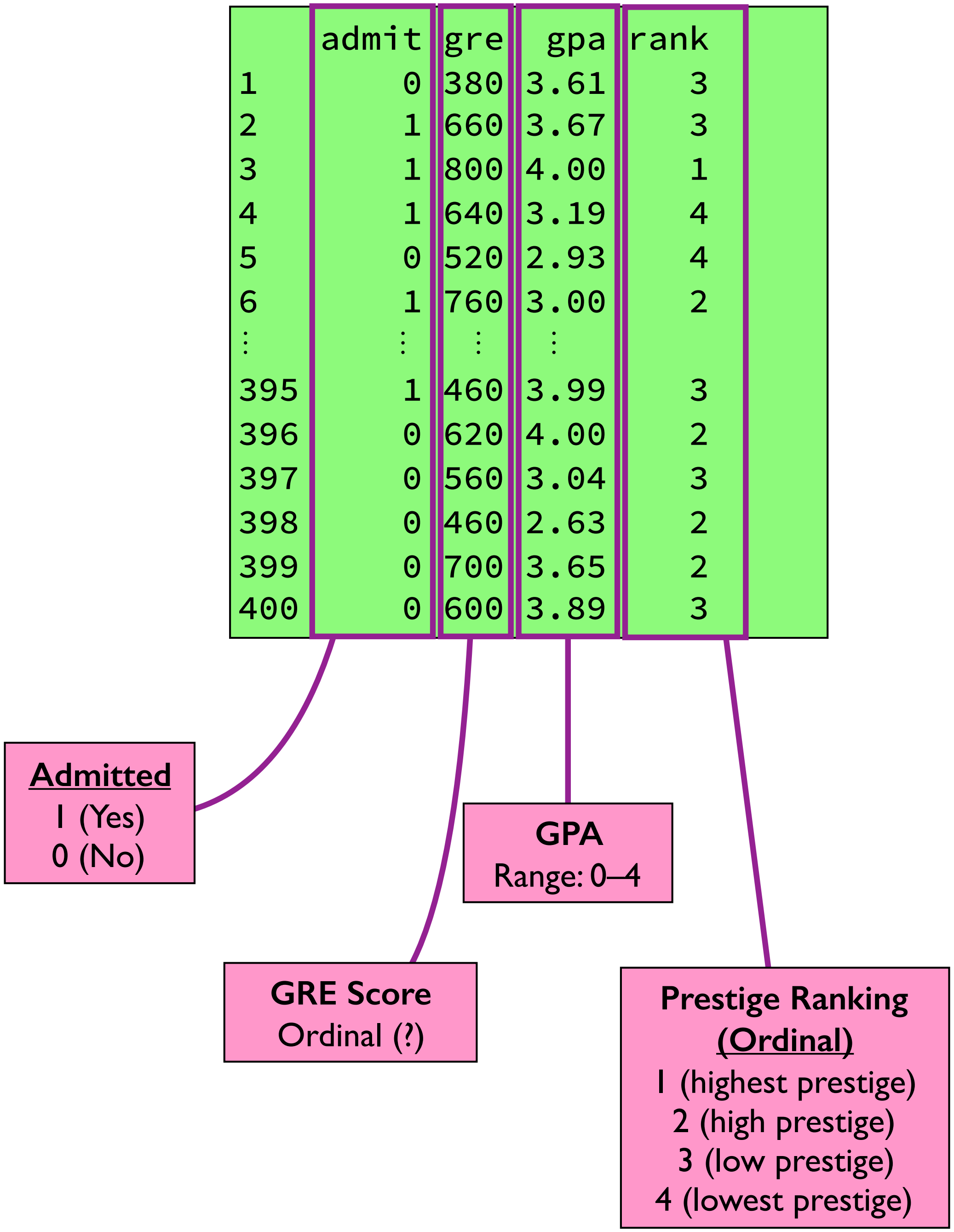
Andrew Zieffler

Preparing the Data for Analysis

```r
# Read in the data
grad <- read.csv(file = "http://www.tc.umn.edu/~zief0002/Data/grad.csv")


# Create new admission variable that is a factor
grad$admit2 <- as.factor(grad$admit)
levels(grad$admit2) <- c("Not Admitted", "Admitted")


# Create new rank variable that is a factor
grad$rank2 <- as.factor(grad$rank, ordered = TRUE)
levels(grad$rank2) <- c("Highest", "High", "Low", "Lowest")
```

Research Question

How do variables, such as Graduate Record Exam scores, grade point average, and prestige of the undergraduate institution, effect admission into graduate school?



```
     admit gre  gpa rank
1        0 380 3.61    3
2        1 660 3.67    3
3        1 800 4.00    1
4        1 640 3.19    4
5        0 520 2.93    4
6        1 760 3.00    2
⋮        ⋮   ⋮    ⋮
395      1 460 3.99    3
396      0 620 4.00    2
397      0 560 3.04    3
398      0 460 2.63    2
399      0 700 3.65    2
400      0 600 3.89    3
```

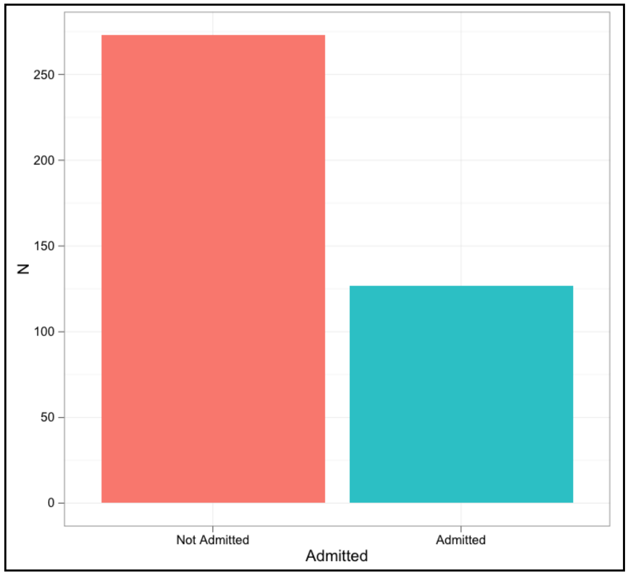
Admitted
1 (Yes)
0 (No)

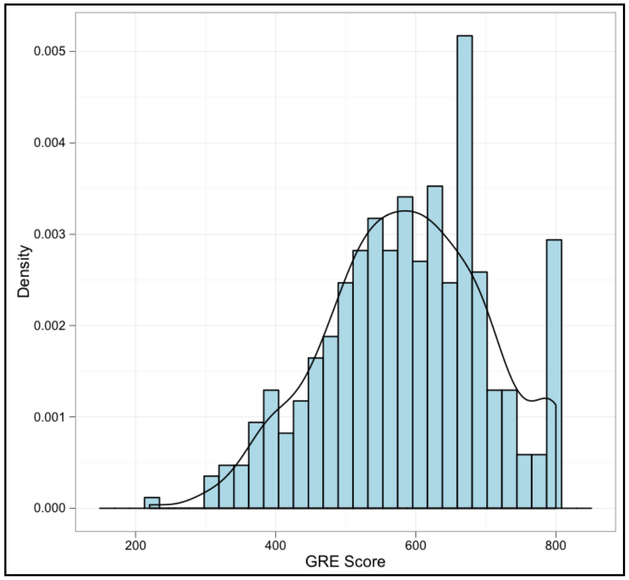
GRE Score
Ordinal (?)

GPA
Range: 0–4

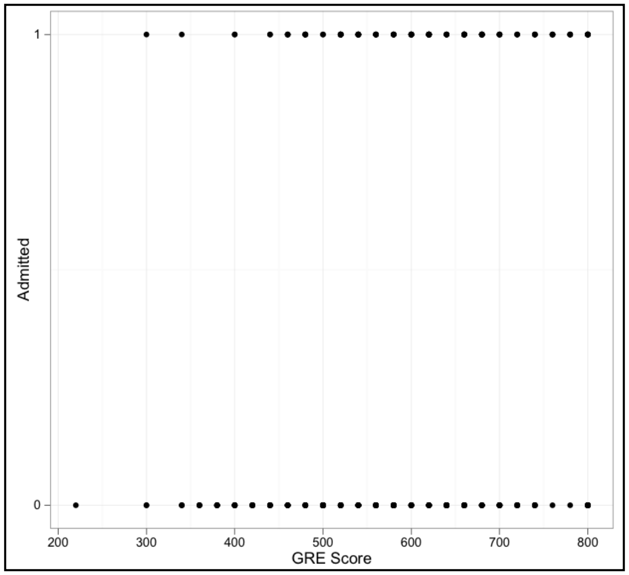
Prestige Ranking
(Ordinal)
1 (highest prestige)
2 (high prestige)
3 (low prestige)
4 (lowest prestige)

```
> summary(grad)

     admit              gre              gpa              rank      
 Min.   :0.0000   Min.   :220.0   Min.   :2.260   Min.   :1.000  
 1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130   1st Qu.:2.000  
 Median :0.0000   Median :580.0   Median :3.395   Median :2.000  
 Mean   :0.3175   Mean   :587.7   Mean   :3.390   Mean   :2.485  
 3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670   3rd Qu.:3.000  
 Max.   :1.0000   Max.   :800.0   Max.   :4.000   Max.   :4.000  
```

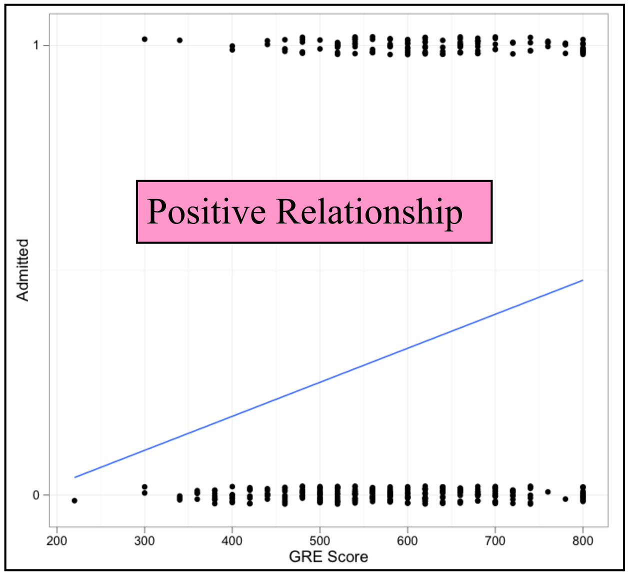




Examine relationship between GRE score and admission.



Problem: Hard to see relationships because of overplotting.
Solution: Add a small amount of random noise to the `admit` variable



Positive Relationship

A correlation between a categorical, dichotomous variable and a quantitative variable is called a **point-biserial** correlation. It is a special case of Pearson's *r* when one of the variables is dummy coded.

```
> cor(grad["admit", "gre"])

          admit       gre
admit 1.0000000 0.1844343
gre   0.1844343 1.0000000
```

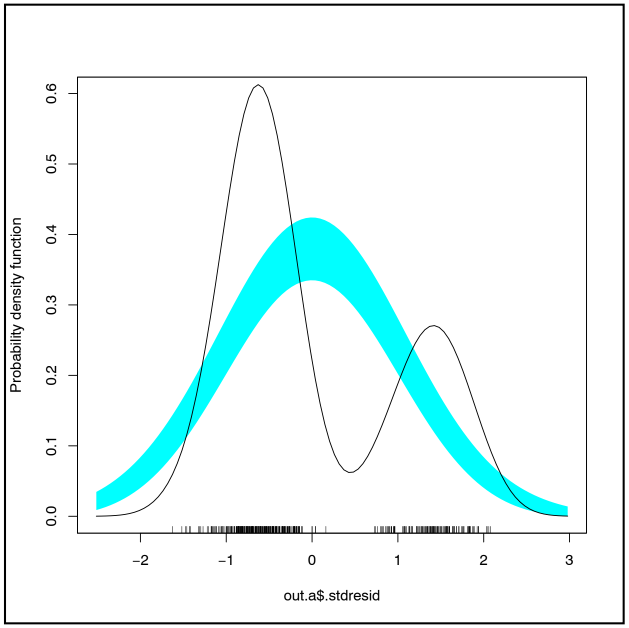
- Students with higher GRE scores are more likely, on average, to be admitted to graduate school.

A correlation between a categorical, dichotomous variable and a ordinal variable is called a **rank-biserial** correlation. This can be obtained by adding the argument `method="spearman"` in the `cor()` function. (In this case they are not that different.)

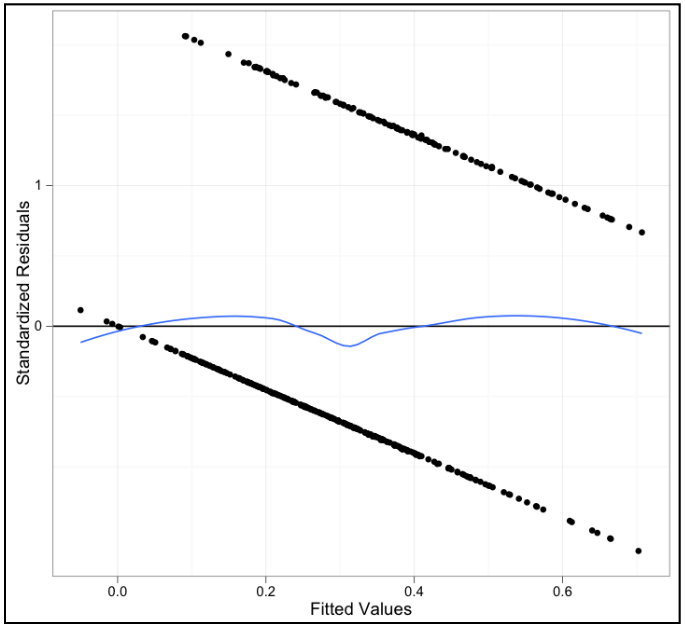
```
> cor(grad["admit", "gre"], method = "spearman")

          admit       gre
admit 1.0000000 0.1766296
gre   0.1766296 1.0000000
```

Fit linear model using all three predictors and examine assumptions.

```
lm.a <- lm(admit ~ gre, data = grad)
```



The errors are not normally distributed.



Two parallel lines

$$\epsilon_i = 1 - \pi_i$$

$$\epsilon_i = -\pi_i$$

Linear model fitted to dichotomous outcomes is called *linear-probability model*

$$E(Y_i) = \beta_0 + \beta_1(X_i) + \epsilon_i$$

Let $\pi_i \equiv P(Y = 1|X = x_i)$ then $E(Y_i) = \pi_i(1) + (1 - \pi_i)(0) = \pi_i$

$$\pi_i = \beta_0 + \beta_1(X_i) + \epsilon_i$$

Re-expression of the model

$$\epsilon_i = Y_i - \hat{Y} = Y_i - \pi_i$$

If $Y_i = 1$ $\quad \epsilon_i = 1 - \pi_i$

If $Y_i = 0$ $\quad \epsilon_i = -\pi_i$

Errors from the linear-probability model are dichotomous...cannot be normally distributed

May (or may not) be a problem depending on sample size...remember the C.L.T.

$$\text{Var}(\epsilon_i) = E(\epsilon_i^2) - (E(\epsilon_i))^2$$

$E(\epsilon_i) = 0$ Assumption of the model

$$\text{Var}(\epsilon_i) = E(\epsilon_i^2)$$

$$= \pi_i(\epsilon_{i|Y_i=1})^2 + (1 - \pi_i)(\epsilon_{i|Y_i=0})^2$$

$$= \pi_i(1 - \pi_i)^2 + (1 - \pi_i)(-\pi_i)^2$$

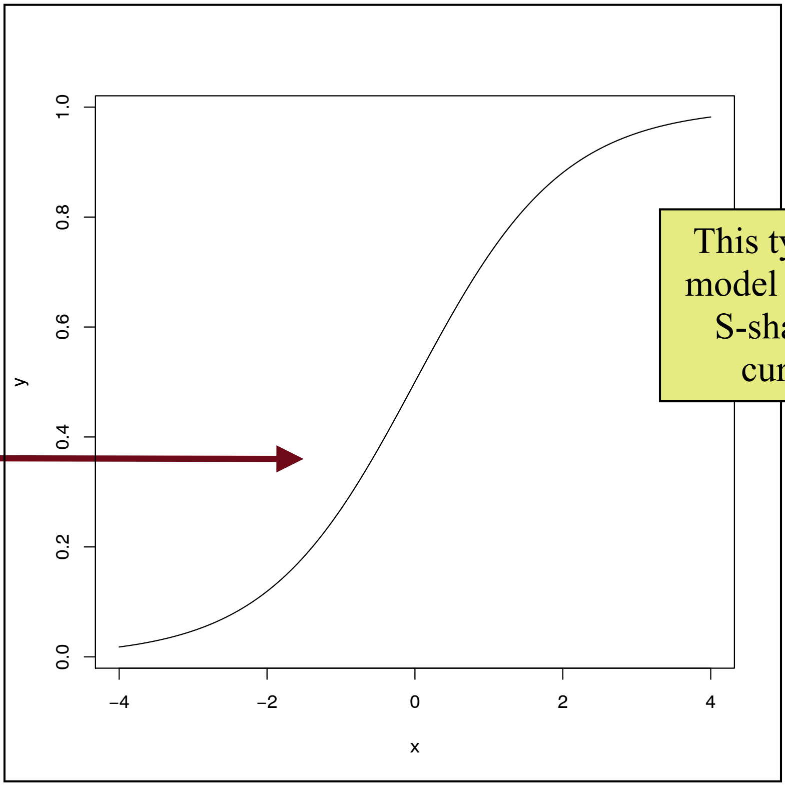
Unless π_i is the same across all x_i, the errors will not have the same variance at each x_i

Furthermore, there is no guarantee that the predicted outcomes will be constrained to the [0, 1] range...this is a problem with linearity (constant rate of change)

Solution: Express the probability $P(Y_i = 1)$ as a non-linear function of X_i

$P(Y_i = 1)$ approaches 0 at slower and slower rates as X_i gets small

$P(Y_i = 1)$ approaches 1 at slower and slower rates as X_i gets large



This type of model has an S-shaped curve

Several key features

- The probability of Y_i never go lower than 0 or above 1
- A person with an X-value of 0 has less probability of Y (e.g., being admitted) than a person with an X-value of 1 (monotonic)
- For really small (or really large) values of X, the probability of Y does not change much (definitely not a constant rate of change)

Transformation of the Probabilities

Probit Transformation

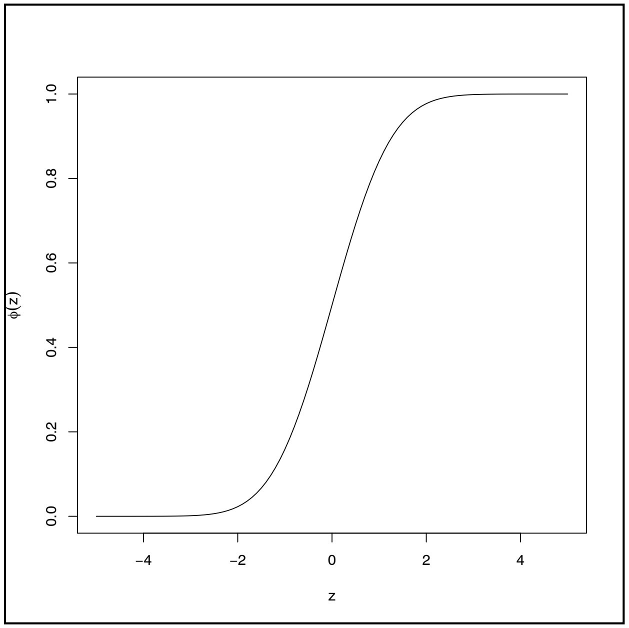
To map the probability $P(Y)$ to the [0, 1] space, we apply a transformation

$$\pi_i = \phi(\beta_0 + \beta_1 X_i)$$

The transformation used is any function that can fit the criteria we had before (monotonic, nonlinear, maps to [0, 1] space)

Common nonlinear model that meets these specifications is the cumulative density function (CDF)—although any CDF would work, here we use the unit normal distribution's CDF

$$\phi(z) = \frac{1}{2\pi}\int_{-\mathrm{inf}}^{z} exp\left(-\frac{1}{2}z^2\right) dz$$



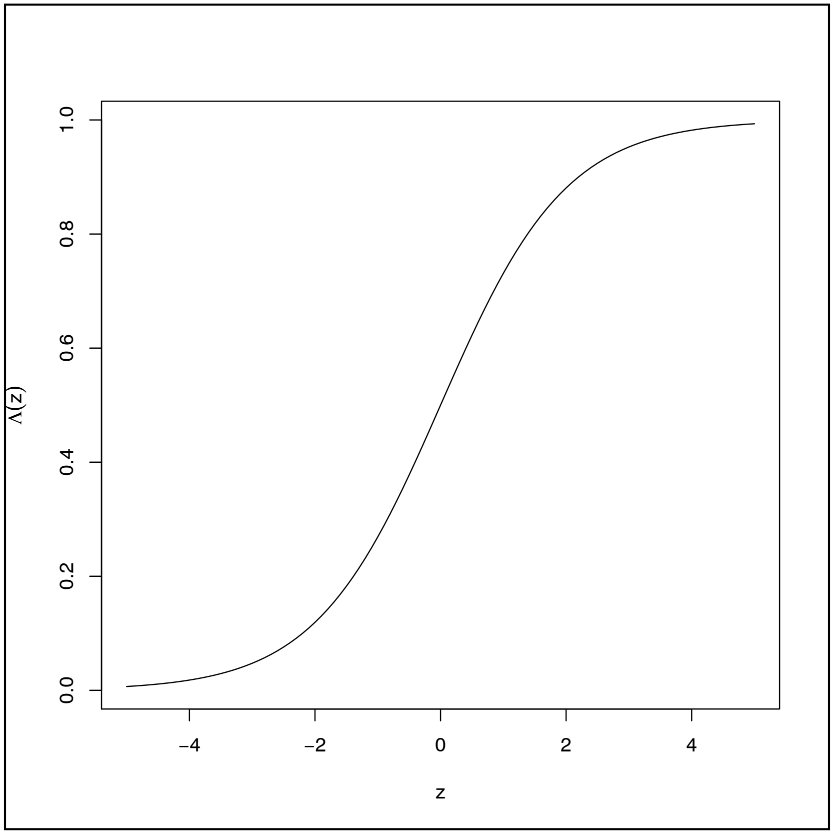
$$\phi(\beta_0 + \beta_1 X_i) = \frac{1}{2\pi}\int_{-\mathrm{inf}}^{\beta_0+\beta_1 X_i} exp\left(-\frac{1}{2}z^2\right) dz$$

This is called a **probit transformation**, and the model is the **linear probit model**.

Logistic Transformation

Another common nonlinear model that meets these specifications is the **logistic** function

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



We map the probability onto the logistic function

$$\pi_i = \Lambda(\beta_0 + \beta_1 X_i)$$

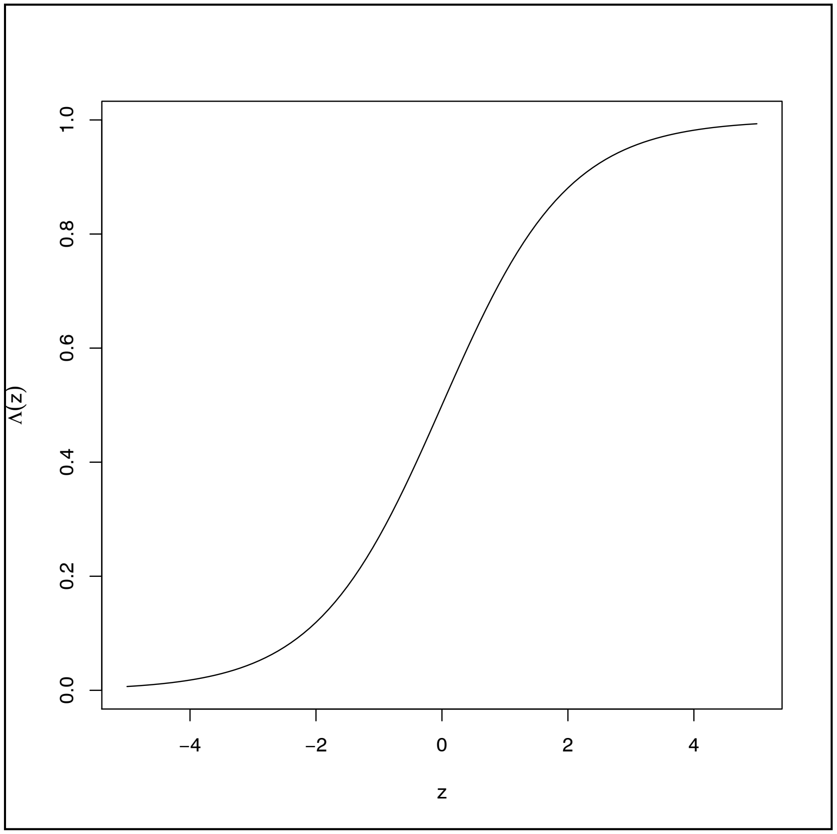
$$\Lambda(\beta_o + \beta_1 X_i) = \frac{1}{1+e^{-(\beta_o+\beta_1 X_i)}}$$

This is called a **logistic transformation**, and the model is the **linear logistic model**.

In practice (once the variances are equalized) there is **no difference** between the probit and logit transformations

Two advantages to using the logit transformation

1. Equation for logit transformation is **simpler** than for probit transformation
2. When we back transform, the inverse of the logit transformation is directly **interpretable**. (The inverse of the probit transformation is not interpretable directly.)



Re-expressing the linear logistic regression

$$\pi_i = \frac{1}{1 + e^{-(\beta_o + \beta_1 X_i)}}$$

$$\frac{\pi_i}{1 - \pi_i} = exp(\beta_o + \beta_1 X_i)$$

This ratio, $\dfrac{\pi_i}{1 - \pi_i}$

is called the **odds.** Odds refers to the relative chances of an occurrence, Y_i

Taking the natural logarithm of both sides of the equation

$$ln\left(\frac{\pi_1}{1 - \pi_i}\right) = \beta_o + \beta_1 X_i$$

The inverse transformation of the probability is called the **logit**.

Logits

Logits are log-odds. It is the log transformed odds that Y_i =1 rather than 0

Another way to think about logistic regression, is to fit a linear model on the log-odds (logit) of the outcome variable Y

$$\frac{\pi_1}{1-\pi_i} = exp(\beta_o + \beta_1 X_i)$$

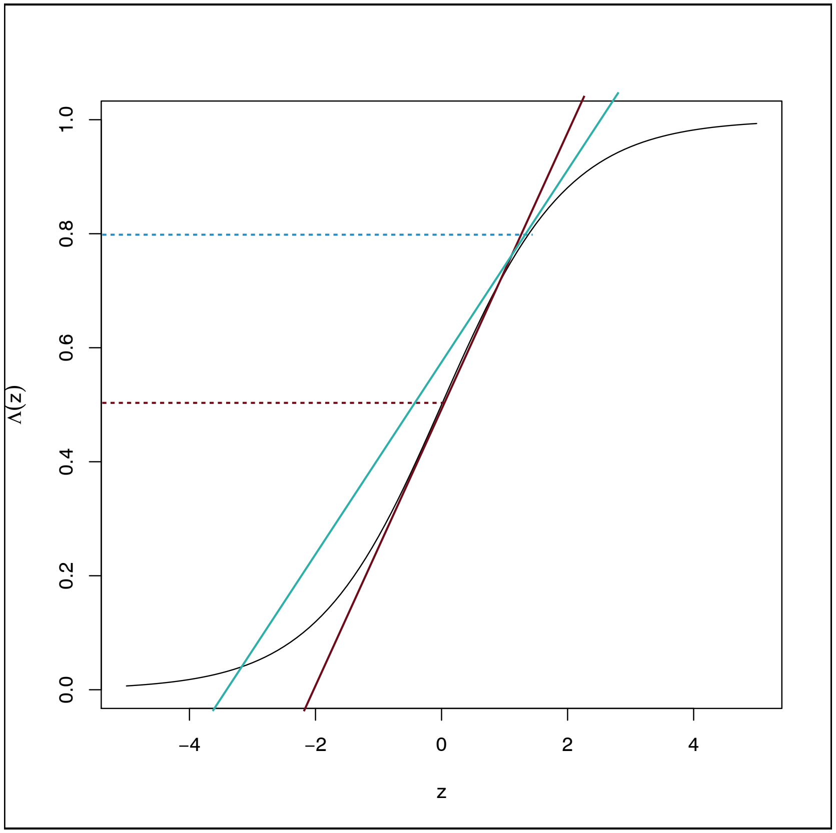
$$= exp(\beta_o) \cdot exp(\beta_1 X_i)$$

$$= exp(\beta_o) \cdot exp(\beta_1)^{X_i}$$

Increasing X_i by one unit changes the logit, on average by β_1 and changes the odds by a factor of $e^{\beta 1}$

Probability	Odds	Logit	
π_i	$\frac{\pi_i}{1-\pi_i}$	$ln\left(\frac{\pi_i}{1-\pi_i}\right)$	
0.01798621	0.01831564	–4	+1
0.04742587	0.04978707	–3	
⋮	⋮	⋮	
0.5	1	0	+1
0.7310586	2.718282	1	
⋮	⋮	⋮	
0.9525741	20.08554	3	+1
0.9820138	54.59815	4	

Consider the slope of the tangent line of the logistic function at a single probability (*Y*-value). This will help us understand the relationship between π_i and X_i



$$\frac{\partial \pi_i}{\partial \beta_1} = \beta_1 \cdot \pi_i \cdot (1 - \pi_i)$$

The tangent line at $\pi_i = 0.8$ is less steep (lower linear rate of change) than the tangent line at $\pi_i = 0.5$.

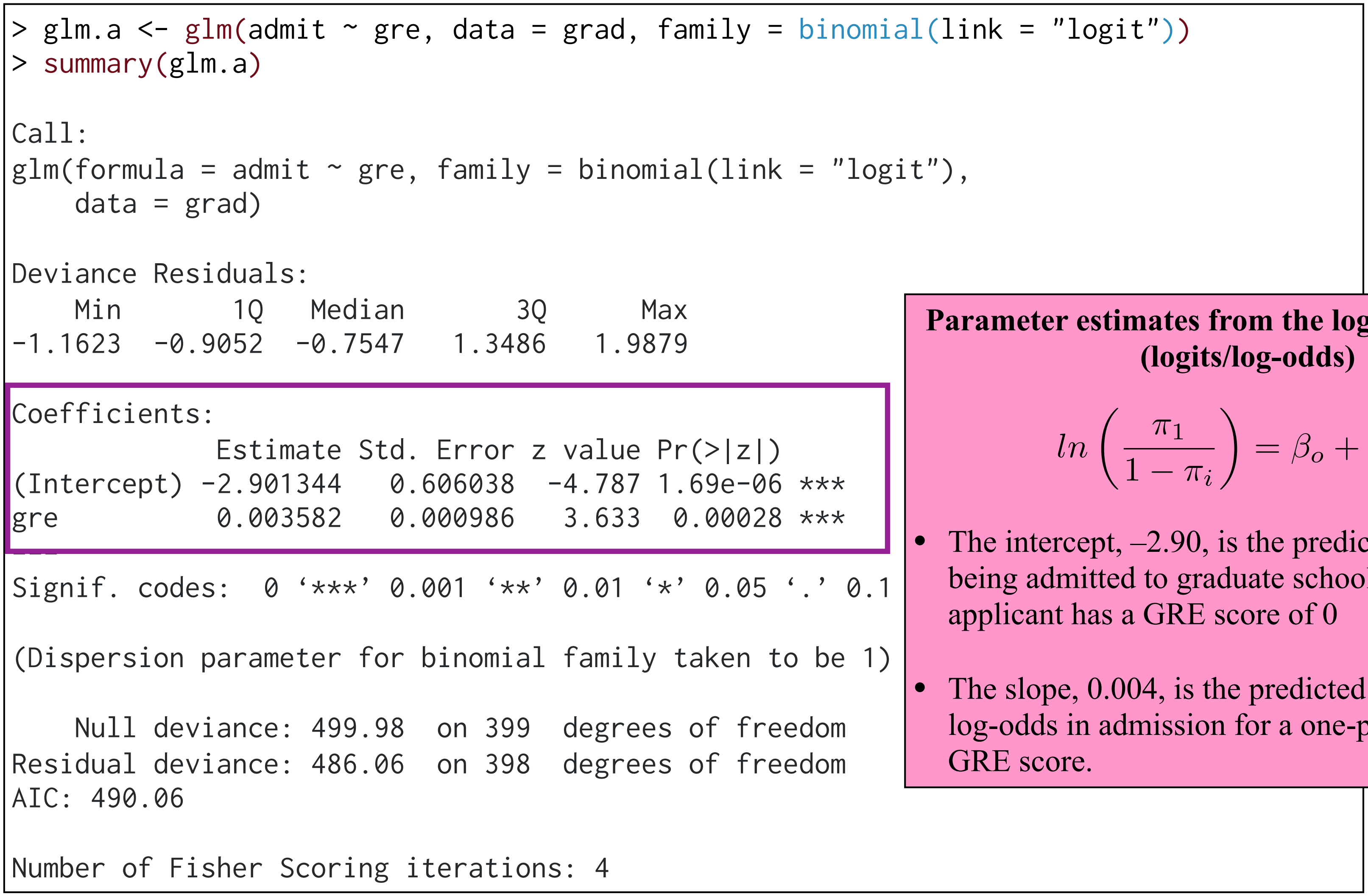
- This suggests the rate of change in probability of *Y* is different depending on the level of *X*
- The slope of these tangent lines can be computed by determining the partial derivative of the logistic function

π_i	$\beta_1 \pi_i (1 - \pi_i)$
0.01	$\beta_1 \times 0.0099$
0.05	$\beta_1 \times 0.0475$
0.1	$\beta_1 \times 0.09$
0.2	$\beta_1 \times 0.16$
0.5	$\beta_1 \times 0.25$
0.8	$\beta_1 \times 0.16$
0.9	$\beta_1 \times 0.09$
0.99	$\beta_1 \times 0.0099$

To fit the logistic model, we use the `glm()` function.

- The `family=` argument specifies the error distribution. For binary outcomes this is a **binomial distribution**.
- Within the `binomial()` function, we then specify the specific transformation via the `link=` argument.

Increasing X_i by one unit changes the logit, on average by β_1 and changes the odds by a factor of $e^{\beta 1}$



```
> glm.a <- glm(admit ~ gre, data = grad, family = binomial(link = "logit"))
> summary(glm.a)

Call:
glm(formula = admit ~ gre, family = binomial(link = "logit"),
    data = grad)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.1623  -0.9052  -0.7547   1.3486   1.9879

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.901344   0.606038  -4.787 1.69e-06 ***
gre          0.003582   0.000986   3.633  0.00028 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 486.06  on 398  degrees of freedom
AIC: 490.06

Number of Fisher Scoring iterations: 4
```

Parameter estimates from the logistic regression (logits/log-odds)

$$ln\left(\frac{\pi_1}{1-\pi_i}\right) = \beta_o + \beta_1 X_i$$

- The intercept, –2.90, is the predicted log-odds for being admitted to graduate school when a applicant has a GRE score of 0
- The slope, 0.004, is the predicted change in the log-odds in admission for a one-point change in GRE score.

Parameter estimates from the logistic regression (odds)

$$\frac{\pi_i}{1-\pi_i} = exp(\beta_o) \cdot exp(\beta_1)^{X_i}$$

- The intercept, $e^{-2.90} = 0.055$, is the predicted odds for being admitted to graduate school when a applicant has a GRE score of 0.
  - ✓ Recall odds values below 1 correspond to odds with higher numbers in the denominator (higher probability of not being admitted) than in the numerator.
  - ✓ The reciprocal, 1/0.055 = 18.199, gives the odds of not being admitted for an applicant with a GRE score of 0.
- The slope, $e^{0.004} = 1.004$, is the predicted change in the odds in admission for a one-point change in GRE score.
  - ✓ A one-point change in GRE score is associated with an increase in the odds of being admitted to graduate school by a factor 1.004

Parameter estimates from the logistic regression (probabilities)

- The intercept, $e^{-2.90}/\ (1 + e^{-2.90}) = 0.052$, is the predicted probability of being admitted to graduate school when a applicant has a GRE score of 0.
  - ✓ The probability of not being admitted for the same GRE score is 0.948.
- The slope is not directly interpretable in terms of the predicted change in probability
  - ✓ Recall the change in probability is a function of the probability (non-linear change), namely $\beta_1\pi_i(1-\pi_i)$
  - ✓ Thus, the predicted change in probability (effect of GRE) is dependent on where the applicant is on the GRE spectrum

Re-expressing the Joint Probability

The joint probability, assuming independence can be expressed as

$$p(y_1, y_2, \ldots, y_n) = \prod_{i=1}^{n} p(y_i)$$

Substituting the probability for a binary variable we get,

$$= \prod_{i=1}^{n} \pi_i^{y_i} (1 - \pi_i)^{1-y_i}$$

Mathematically,

$$X^{1-c} = \frac{X^1}{X^c}$$

Using this, we can re-write the product as...

$$= \prod_{i=1}^{n} \pi_i^{y_i} \frac{(1 - \pi_i)^1}{(1 - \pi_i)^{y_i}}$$

$$= \prod_{i=1}^{n} \frac{\pi_i^{y_i}}{(1 - \pi_i)^{y_i}} (1 - \pi_i)$$

Re-organize

$$= \prod_{i=1}^{n} \left[\frac{\pi_i}{(1 - \pi_i)} \right]^{y_i} (1 - \pi_i)$$

Simplify

From the model, recall

$$\frac{\pi_i}{(1-\pi_i)} = e^{\beta_0+\beta_1(X_{i1})}$$

We can use this equation and solve for both π_i and $(1 - \pi_i)$

$$\begin{aligned}
\pi_i &= e^{\beta_0+\beta_1(X_{i1})}(1-\pi_i) \\
&= e^{\beta_0+\beta_1(X_{i1})} - \pi_i e^{\beta_0+\beta_1(X_{i1})} \\
\pi_i + \pi_i e^{\beta_0+\beta_1(X_{i1})} &= e^{\beta_0+\beta_1(X_{i1})} \\
\pi_i \left(1 + e^{\beta_0+\beta_1(X_{i1})}\right) &= e^{\beta_0+\beta_1(X_{i1})} \\
\pi_i &= \boxed{\frac{e^{\beta_0+\beta_1(X_{i1})}}{1+e^{\beta_0+\beta_1(X_{i1})}}}
\end{aligned}$$

Solving for π_i

$$1 - \pi_i = \boxed{\pi_i}\frac{1}{e^{\beta_0+\beta_1(X_{i1})}}$$

Solving for $(1 - \pi_i)$

We now substitute the value of π_i into this equation

$$\begin{aligned}
1 - \pi_i &= \boxed{\frac{e^{\beta_0+\beta_1(X_i)}}{1+e^{\beta_0+\beta_1(X_{i1})}}}\frac{1}{e^{\beta_0+\beta_1(X_{i1})}} \\
&= \frac{1}{1+e^{\beta_0+\beta_1(X_{i1})}}
\end{aligned}$$

Substitute

Simplify

Maximum Likelihood Estimation for the Logistic Regression Model

Thus the joint probability can be expressed as...

$$p(y_1, y_2, \ldots, y_n) = \prod_{i=1}^{n} \left[e^{\beta_0+\beta_1(X_{i1})}\right]^{y_i} \left[\frac{1}{1+e^{\beta_0+\beta_1(X_i)}}\right]$$

If we treat the observed binary response vector
$y_i = (y_1, y_2, \ldots, y_n)$
as fixed (after all it is what we observed), then this becomes a function of the parameters β_0 and β_1 referred to as the **likelihood function** which is written as...

$$\mathcal{L}(\beta_0, \beta_1, \ldots, \beta_k) = \prod_{i=1}^{n} \left[e^{\beta_0+\beta_1(X_{i1})+\ldots+\beta_k(X_{ik})}\right]^{y_i} \left[\frac{1}{1+e^{\beta_0+\beta_1(X_{i1}+\ldots+\beta_k(X_{ik}))}}\right]$$

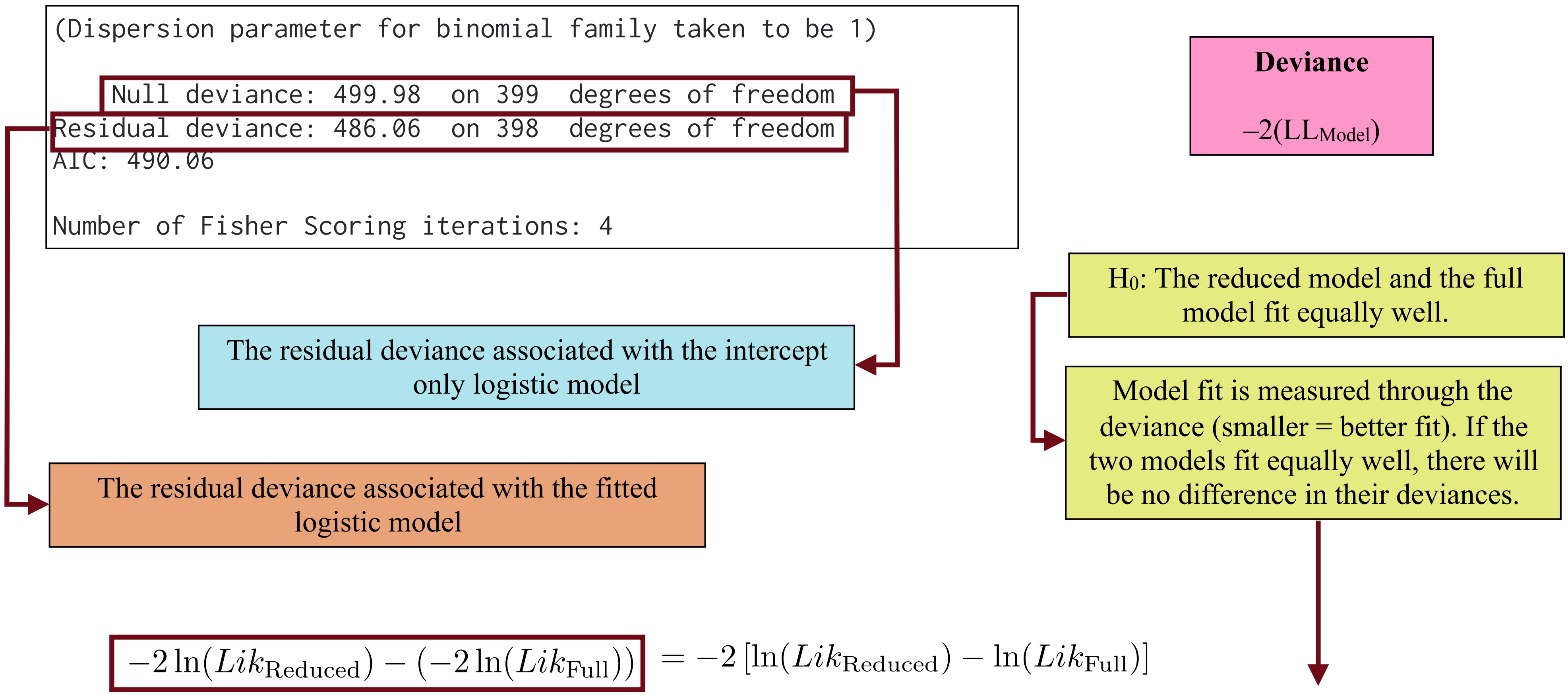
Goal: Find the values

$$\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_k$$

that maximize the likelihood.

Inference for the Logistic Regression Model

Model-Level Inference



```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 486.06  on 398  degrees of freedom
AIC: 490.06

Number of Fisher Scoring iterations: 4
```

Deviance

$-2(LL_{\text{Model}})$

The residual deviance associated with the intercept only logistic model

The residual deviance associated with the fitted logistic model

H_0: The reduced model and the full model fit equally well.

Model fit is measured through the deviance (smaller = better fit). If the two models fit equally well, there will be no difference in their deviances.

$$\boxed{-2\ln(Lik_{\text{Reduced}}) - (-2\ln(Lik_{\text{Full}}))} = -2\left[\ln(Lik_{\text{Reduced}}) - \ln(Lik_{\text{Full}})\right]$$

Difference in deviances
(Likelihood ratio)

$$= -2\ln\left(\frac{Lik_{\text{Reduced}}}{Lik_{\text{Full}}}\right)$$

$$499.98 - 486.06 = 13.92$$

Sample evidence against H_0

If n is sufficiently large, the likelihood ratio is $\sim\chi^2\ (df_{\text{Full}} - df_{\text{Reduced}})$

```
> anova(glm.a, test = "Chisq")
     Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                   399     499.98
gre   1    13.92       398     486.06 0.0001907 ***
```

Inference for the Logistic Regression Model

Parameter-Level Inference

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)    
(Intercept) -2.901344   0.606038  -4.787 1.69e-06 ***
gre          0.003582   0.000986   3.633  0.00028 ***
```

Null hypothesis associated with each parameter

$$H_0 : \beta_j = 0$$

To examine the degree of statistical evidence against the null hypothesis, we carry out a **Wald test**.

$$Z = \frac{\hat{\beta}_j}{SE_{\hat{\beta}_j}}$$

$Z_j \sim \mathscr{N}(0, 1)$

$3.633^2 = 13.19 \approx 13.92$

When there is only one parameter that is different between a full and the reduced model, for large *n* the likelihood ratio (LR) = Z^2_{Wald}

Confidence intervals can also be computed based on the Wald statistic

$$\hat{\beta}_j \pm 2 \times SE_{\hat{\beta}_j}$$

$$0.003582 \pm 2 \times 0.000986$$

$$[0.00161, 0.005554]$$

Confidence limits for the effect on the logit

Most statisticians prefer to give confidence limits for the odds ratio rather than the logit.

$$\left[e^{0.00161}, e^{0.005554}\right]$$

$$[1.001611, 1.005569]$$

Confidence limits for the effect on the odds ratio

Better Confidence Intervals

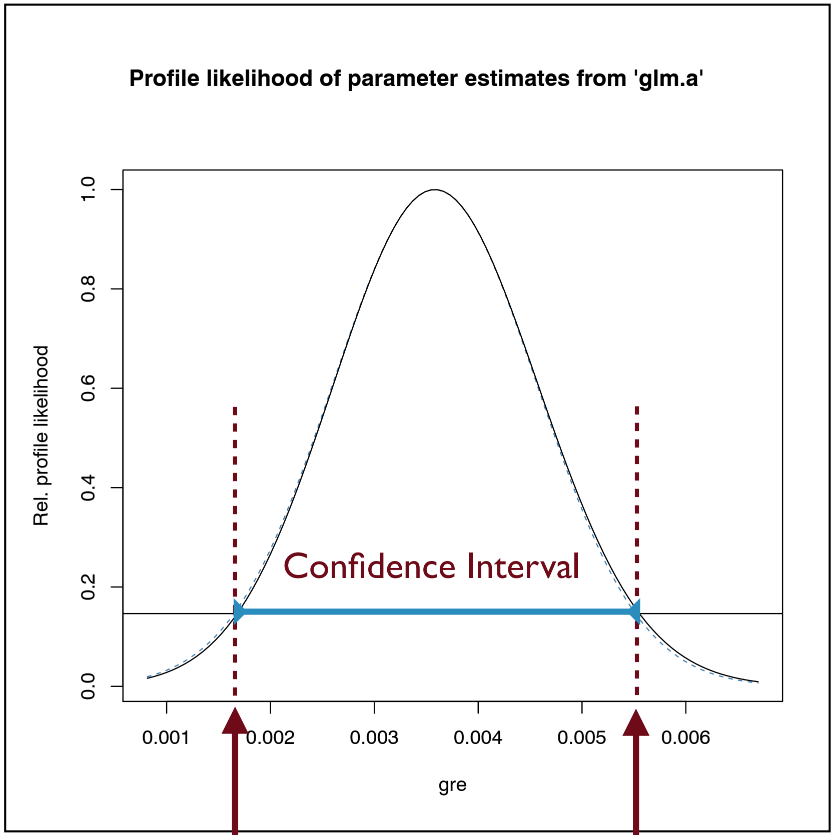
Profile Likelihood CIs

Confidence intervals based on the Wald statistic are formed by **inverting** the Wald tests

- Inversion determines the parameter values for which the null hypothesis is not rejected
- For small sample sizes the Wald test can be terribly wrong , which means the CI based on the inversion will be wrong as well

Better CIs can be produced by inverting the LR test. This is done by **profiling the likelihood**.

- A set of potential parameter values are chosen
- For this set of values, the likelihood is maximized over the other parameters included in the model
- The result is plotted in a profile plot



```
> confint(glm.a)
Waiting for profiling to be done...
                   2.5 %       97.5 %
(Intercept) -4.119988259 -1.739756286
gre          0.001679963  0.005552748
```

Plotting the Fitted Model

Step 1: Create a data frame of the predictors in the model.

```
> plotdata <- data.frame(
     gre = seq(from = 200, to = 800, by = 20)
     )
```

model

data to predict from

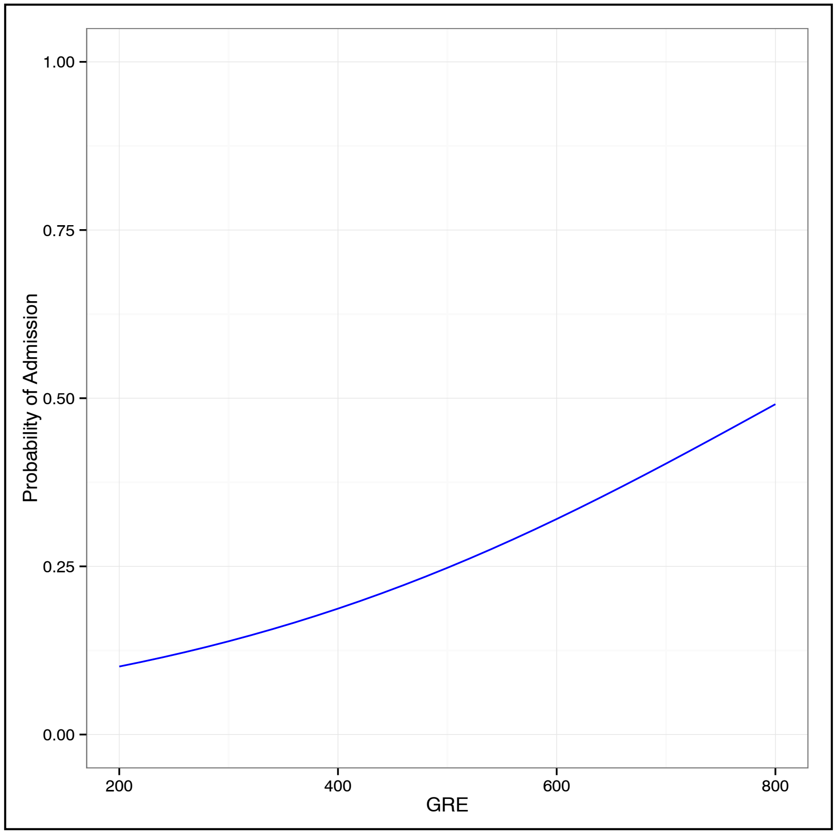
unit of predictions "response" means probabilities

Step 2: Use the `predict()` function to produce a fitted value for each row in the newly created data frame.

```
> plotdata$fitted <- predict(glm.a, newdata = plotdata, type = "response")
```

Step 3: Plot the fitted values versus the *X*s

```
> ggplot(data = plotdata, aes(x = gre, y = fitted)) +
     geom_line(color = "blue") +
     xlab("GRE") +
     ylab("Probability of Admission") +
     ylim(c(0, 1)) +
     theme_bw()
```



Shortcut when there is only one predictor

```
> ggplot(data = grad, aes(x = gre, y = admit)) +
    geom_smooth(method = "glm", family = "binomial) +
    xlab("GRE") +
    ylab("Probability of Admission") +
    ylim(c(0, 1)) +
    theme_bw()
```

Adding a Confidence Envelope to the Model

Use the `se.fit = TRUE` argument in the `predict()` function

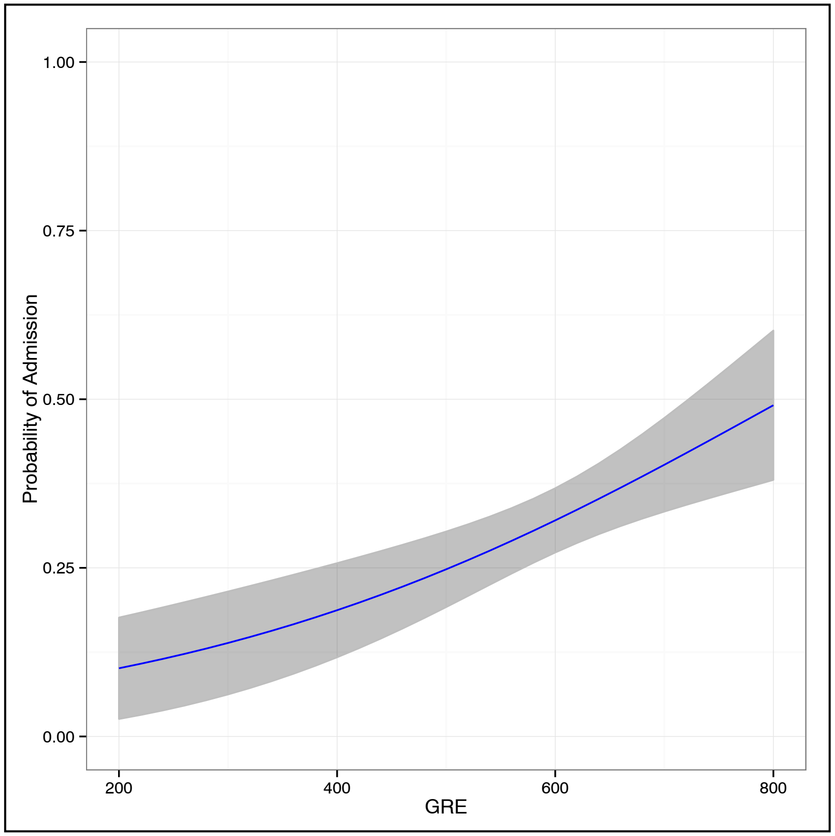
```
> fitted <- predict(glm.a, newdata = plotdata,
     type = "response", se.fit = TRUE)
```

Create a data frame that includes the predictor, the fitted values and the SEs

```
> plotdata2 <- data.frame(
     gre = plotdata$gre,
     fitted = fitted$fit,
     se = fitted$se.fit
     )
```

Compute the lower and upper bounds for the confidence envelope

```
> plotdata2$lowerLimit <- plotdata2$fitted - 2 * plotdata2$se
> plotdata2$upperLimit <- plotdata2$fitted + 2 * plotdata2$se
```



Use geom_ribbon() to draw the confidence envelope

```
> ggplot(data = plotdata2, aes(x = gre, y = fitted)) +
     geom_ribbon(aes(ymin = lowerLimit, ymax = upperLimit),
          color = "grey", alpha = 0.3) +
     geom_line(color = "blue") +
     xlab("GRE") +
     ylab("Probability of Admission") +
     ylim(c(0, 1)) +
     theme_bw()
```